(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開平6－52161
(43)公開日　平成６年(1994)２月25日

(51)Int.Cl.[5]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　15/20	５５０　Ｅ	9288－5L		

審査請求　未請求　請求項の数10(全 32 頁)

(21)出願番号　特願平4－206722

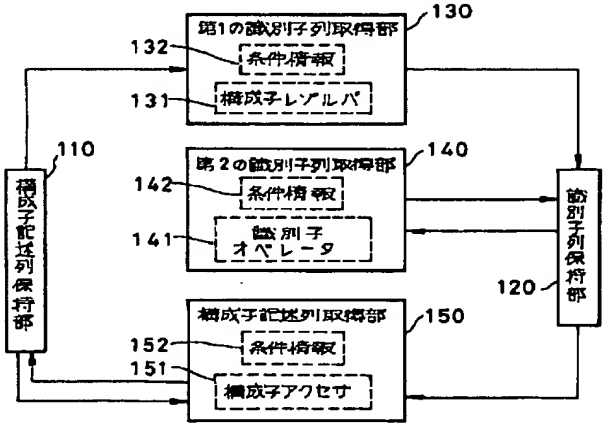
(22)出願日　平成４年(1992)８月３日

(71)出願人　000005496
富士ゼロックス株式会社
東京都港区赤坂三丁目３番５号
(72)発明者　中津山　恒
神奈川県横浜市保土ケ谷区神戸町134番地
横浜ビジネスパーク　イーストタワー
富士ゼロックス株式会社内
(72)発明者　楠本　浩二
神奈川県横浜市保土ケ谷区神戸町134番地
横浜ビジネスパーク　イーストタワー
富士ゼロックス株式会社内
(74)代理人　弁理士　木村　高久

最終頁に続く

(54)【発明の名称】　文書処理方法及び文書処理装置

(57)【要約】
【目的】他の文書処理機能と組合わせて、文書或いは文書部品の検索、挿入、合成、更新を正確に処理することのできる文書処理方法及び文書処理装置を提供する。
【構成】構成子記述列保持部１１０には、予め記述された文書の内容である構成子記述列群が格納されている。第１の識別子列取得部１３０は、指定された第１の条件に適合する構成子に対応する識別子列を、構成子記述列保持部１１０内の構成子記述列群から取得する。この取得結果である識別子列は、識別子列保持部１２０に保持される。第２の識別子列取部１４０は、識別子列保持部１２０に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する。また取得した識別子列を、既に保持されている他の識別子列と区別できるように識別子列保持部１２０に格納する。構成子記述列取得部１５０は、識別子列保持部１２０に保持されている第２の識別子列取得部１４０により得られた識別子列に対応する構成子記述列を、構成子記述列保持部１１０内の指定された構成子記述列から取得する。



【特許請求の範囲】

【請求項１】グラフ理論におけるグラフによって表現される文書に対する所定の処理を行う文書処理方法において、

予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別予列を取得するステップと、

前記識別子列から、指定された第２の条件に適合する識別子列を取得するステップと、

前記第２の条件に適合する識別子列に基づいて、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に対する所定の処理を実行するステップとを含むことを特徴とする文書処理方法。

【請求項２】前記所定の処理を実行するステップは、

前記第２の条件に適合する識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列から取得するステップ、

或いは前記第２の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入するステップ、

或いは前記第２の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って変更するステップ、

或いは前記第２の条件に適合する識別子列を検査し、この検査結果が指定された第３の条件に適合する場合のみに、当該識別子列に対応する構成子群を有する前記予め記述された単数又は複数の文書の文書名を出力するステップのうち、いずれかのステップ又は複数のステップを含むことを特徴とする請求項１記載の文書処理方法。

【請求項３】グラフ理論におけるグラフによって表現される文書の部品を検索する文書処理装置において、

予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、

前記第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、

前記識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する第２の識別子列取得手段と、

前記第２の識別子列取得手段により得られた識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列から取得する構成子記述列取得手段とを具えたことを特徴とする文書処理装置。

【請求項４】グラフ理論におけるグラフによって表現される文書の部品を所定の文書に挿入する文書処理装置において、

予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、

前記第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、

前記識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する第２の識別子列取得手段と、

前記第２の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入する構成子記述列挿入手段とを具えたことを特徴とする文書処理装置。

【請求項５】グラフ理論におけるグラフによって表現される文書の部品の属性値を更新する文書処理装置において、

予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、

前記第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、

前記識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する第２の識別子列取得手段と、

前記第２の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段とを具えたことを特徴とする文書処理装置。

【請求項６】グラフ理論におけるグラフによって表現される文書を検索する文書処理装置において、

予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、

前記第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、

前記識別子列保持手段に保持されている識別子列から、指定された第２の条件を満たす識別子列を取得する第２の識別子列取得手段と、

前記第２の識別子列取得手段により得られた識別子列を

検査し、この結果が指定された第３の条件に適合する場合は、当該識別子列に対応する構成子群を有する前記予め記述された単数又は複数の文書の文書名を出力する文書名出力手段とを具えたことを特徴とする文書処理装置。

【請求項７】前記構成子記述列取得手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入する構成子記述列挿入手段を更に具えたことを特徴とする請求項３記載の文書処理装置。

【請求項８】前記構成子記述列取得手段により得られた構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段を更に具えたことを特徴とする請求項３記載の文書処理装置。

【請求項９】グラフ理論におけるグラフによって表現される文書に対する処理を行う文書処理装置において、

予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、

前記第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、

前記識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する第２の識別子列取得手段と、

前記第２の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段と、

前記構成子記述列変更手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する構成子記述列挿入手段とを具えたことを特徴とする文書処理装置。

【請求項１０】グラフ理論におけるグラフによって表現される文書に対する処理を行う文書処理装置において、

予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、

前記第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、

前記識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する第２の識別子列取得手段と、

前記予め記述された単数又は複数の文書の構成子記述列から、前記第２の識別子列取得手段により得られた識別子列に対応する構成子記述列を取得する構成子記述列取得手段と、

前記構成子記述列取得手により得られた構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段と、

前記構成子記述列変更手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する構成子記述列挿入手段とを具えたことを特徴とする文書処理装置。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】この発明は、グラフ理論におけるグラフによって表現される文書を処理する文書処理方法及び文書処理装置に関する。

【０００２】

【従来の技術野】電子文書の論理構造をグラフ（木や一般のネットワーク）によって表すことは従来から広く行われている。例えば国際規格ＯＤＡ（ISO/IS 8613, InformationProcessing-Text and Office System-Office Document Architecture(ODA) andInterchange Format(1988)）では、文書の論理構造をノード（オブジェクト）によって表現している。またハイパーテキストでは、ノードとリンクとで構成するネットワークが文書である。

【０００３】論理構造を利用して、記憶装置に記憶された文書を対象として、文書や文書部品の検索、挿入、合成、更新を行う文書処理装置が実現されている。このような文書処理装置を用いて、文書や文書部品の検索、挿入、合成、更新を行うには、通常、問合せ言語を用いて条件や動作を指定する。

【０００４】文書処理装置においては、何等かの手段によって、文書中のノードのうち処理の対象となるものを特定する必要がある。

【０００５】この処理の対象となるものを特定する文書処理装置としては、本願出願人によって出願された特願平２－２３５７４０号（発明の名称：電子文書生成方式）、特願平４－１５４６３号（発明の名称：文書データベース装置）に開示されたものがある。

【０００６】特願平２－２３５７４０号に開示されたものにおいては、図１８に示す様な論文の論理構造を有する構造化文書である入力文書群から出力文書を得るために、以下に述べる様なＬＩＳＰ言語に似た構造を有する手続き的な操作言語によって記述されたプログラムを実行するようにしている。

【０００７】文書プログラム識別子：論文抄録作成

```
(SetID “論文抄録作成”
(MakerRoot (MakeNewNode “論文抄録”))
(AddChild 論文抄録 (MakeNewNode “文献１”))
(AddChild 論文抄録 (MakeNewNode “文献２”))
(AddChild 論文抄録 (MakeNewNode “文献３”))
(AddSubTree 文献１ 表題@論文Ａ)
```

(AddSubTree　文献１　著者名@論文Ａ)
(AddSubTree　文献１　所属@論文Ａ)
(AddSubTree　文献１　要約@論文Ａ)
(AddSubTree　文献２　表題@論文Ｂ)
(AddSubTree　文献２　著者名@論文Ｂ)
(AddSubTree　文献２　所属@論文Ｂ)
(AddSubTree　文献２　要約@論文Ｂ)
(AddSubTree　文献３　表題@論文Ｃ)
(AddSubTree　文献３　著者名@論文Ｃ)
(AddSubTree　文献３　所属@論文Ｃ)
(AddSubTree　文献３　要約@論文Ｃ)
(ChangeText　“えっくす研究所” “Our laboratory”)
(SetAttribute“編集者” “Auto”)
(SetAttribute“編集日” (Date))

図１８に示す様な入力文書に対し、このようなプログラムを実行することにより、図１９に示す様な出力文書が得られる。

【０００８】一方、特願平４－１５４６３号に開示されたものにおいては、図２０に示すような論文の論理構造を有する構造化文書からの文書部品の抽出のために、以下に述べるようなＳＱＬ言語に似た問合せ言語によって記述されたプログラムを実行するようにしている。

【０００９】

```
Project
    Project
        タイトル
        著者名
        Project
            段落タイトル
        Collapse
        From 段落
    From *
From ／データベース関連研究／論文
```

図２０に示す様な文書に対し、このようなプログラムを実行することにより、図２１に示すような結果が得られる。この文書による検索結果は文書あるいは文書部品の並びである。

【００１０】これらの公報に開示されたものでは、ノードの属性値についての条件を指定することによって、処理の対象となるノードを特定する方法をとっている。この方法では、例えば、「“著者”というタグが付随している」という条件の指定を行って、処理の対象となるノードを特定する。

【００１１】文書部品の検索は、上記の様な指定によって特定されたノードを、文書中から抜き出すことである。また文書部品の属性更新は、特定されたノードに属性を追加、削除、変更することである。さらに文書部品の挿入は、特定されたノードからの相対位置（例えば直後）を指定して、指定されたノードを挿入することである。

【００１２】

【発明が解決しようとする課題】しかしながら、上記従来の各公報に開示されたものでは、文書あるいは文書部品の操作は、全て操作言語で記述されている必要がある。従って、文書部品の抽出、構造の変更、属性の設定、テキストの変更といった一連の処理は、一つの処理系で一括して行われる。

【００１３】これらの処理方式においては、ひとつの処理を行う際に、操作言語の処理系である文書処理機能（装置）と他の文書処理機能（装置）とを連携させ、きめ細かな処理を行う事は極めて難しい。

【００１４】以上の問題点を具体例を用いて説明する。ノードがテキストで表現されている場合は、汎用のテキスト処理機能（装置）も文書処理機能（装置）として機能し得る。従って、この汎用テキスト処理機能（装置）と、上述した文書処理機能（装置）（上記各公報に開示されたもの）とを組合せて、

（１）文書処理機能によって、操作対象となるノード列を取り出す。

（２）テキスト処理機能によって、ノード列に含まれる文字列の挿入、置換を行う。

（３）文書処理機能によって、上記（２）の処理で得られたノード列を他のノード列に挿入する。

といった処理を行うことが考えられる。

【００１５】ところが、汎用テキスト処理機能と、問合せ言語の処理系である従来の文書処理機能とを組合せて使用することは容易ではない。そのため上記（１）～（３）の処理を分けて実現することができず、以下の問題が生じる。

【００１６】テキスト処理機能における操作対象は、上記（１）の処理で取り出された中で更に限定されたノード列のみであるが、この処理が全て終わったときに得られるノード列全体に対して、上記（２）の処理の操作を行うことになってしまう。このため、本来は変更すべきでないノード列も変更されてしまうことがある。

【００１７】また上記（１）の処理によって文書から操作対象となるノード列を取り出し、上記（２）の処理が終了した後、元の文書に戻そうとしても、各々のノード列がその文書の中で属していた部分を特定することができない。何故なら、ノードの属性値についての条件を指定することによってノード列を特定したので、ノード列が元の文書でどこに属していたか正確にはわからない。このため上記（３）の処理を行うことができない。

【００１８】このように従来の文書処理機能では、他の文書処理機能と組合せてきめ細かな処理を行うことは出来なかった。

【００１９】この発明は、他の文書処理機能と組み合わせて、文書あるいは文書部品の検索、挿入、合成、更新を正確に処理することのできる文書処理方法及び文書処

理装置を提供することを目的とする。

【００２０】

【課題を解決するための手段】第１の発明は、グラフ理論におけるグラフによって表現される文書に対する所定の処理を行う文書処理方法において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得するステップと、前記識別子列から、指定された第２の条件に適合する識別子列を取得するステップと、前記第２の条件に適合する識別子列に基づいて、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に対する所定の処理を実行するステップとを含むことを特徴とする。

【００２１】第２の発明は、第１の発明において、前記所定の処理を実行するステップは、前記第２の条件に適合する識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列から取得するステップ、前記第２の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入するステップ、前記第２の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って変更するステップ、前記第２の条件に適合する識別子列を検査し、この検査結果が指定された第３の条件に適合する場合のみに、当該識別子列に対応する構成子群を有する前記予め記述された単数又は複数の文書の文書名を出力するステップのうち、いずれかのステップ又は複数のステップを含むことを特徴とする。

【００２２】第３の発明は、グラフ理論におけるグラフによって表現される文書の部品を検索する文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、該第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する第２の識別子列取得手段と、該第２の識別子列取得手段により得られた識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列から取得する構成子記述列取得手段とを具えている。

【００２３】第４の発明は、グラフ理論におけるグラフによって表現される文書の部品を所定の文書に挿入する文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、該第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する第２の識別子列取得手段と、該第２の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入する構成子記述列挿入手段とを具えている。

【００２４】第５の発明は、グラフ理論におけるグラフによって表現される文書の部品の属性値を更新する文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、該第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する第２の識別子列取得手段と、該第２の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段とを具えている。

【００２５】第６の発明では、グラフ理論におけるグラフによって表現される文書を検索する文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、該第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第２の条件を満たす識別子列を取得する第２の識別子列取得手段と、該第２の識別子列取得手段により得られた識別子列を検査し、この結果が指定された第３の条件に適合する場合は、当該識別子列に対応する構成子群を有する前記予め記述された単数又は複数の文書の文書名を出力する文書名出力手段とを具えている。

【００２６】第７の発明は、第３の発明において、前記構成子記述列取得手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入する構成子記述列挿入手段を更に具えたことを特徴とする。

【００２７】第８の発明は、第３の発明において、前記構成子記述列取得手段により得られた構成子記述列を、

指定された変更指示情報に従って変更する構成子記述列変更手段を更に具えたこと特徴とする。

【００２８】第９の発明は、グラフ理論におけるグラフによって表現される文書に対する処理を行う文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、該第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する第２の識別子列取得手段と、該第２の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段と、該構成子記述列変更手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する構成子記述列挿入手段とを具えている。

【００２９】第１０の発明は、グラフ理論におけるグラフによって表現される文書に対する処理を行う文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得する第１の識別子列取得手段と、該第１の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得する第２の識別子列取得手段と、前記予め記述された単数又は複数の文書の構成子記述列から、前記第２の識別子列取得手段により得られた識別子列に対応する構成子記述列を取得する構成子記述列取得手段と、該構成子記述列取得手により得られた構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段と、該構成子記述列変更手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する構成子記述列挿入手段とを具えている。

【００３０】

【作用】第１の発明では、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得し、この識別子列から、指定された第２の条件に適合する識別子列を取得し、この第２の条件に適合する識別子列に基づいて、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に対する所定の処理を実行する。

【００３１】第２の発明では、第１の発明において、前記所定の処理をするに際し、前記第２の条件に適合する識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列から取得する処理、前記第２の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入する処理、前記第２の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って変更する処理、前記第２の条件に適合する識別子列を検査し、この検査結果が指定された第３の条件に適合する場合のみに、前記予め記述された単数又は複数の文書の文書名を出力する処理のうち、いずれかの処理又は複数の処理を実行する。

【００３２】第３の発明では、第１の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得し、更にこの識別子列を識別子列保持手段に格納する。第２の識別子列取得手段が、該識別子列保持手段に保持されている識別子列中から、指定された第２の条件に適合する識別子列を取得し、構成子記述列取得手段が、この識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列から取得する。従って、指定された文書から条件に適合する文書部品を検索することができる。

【００３３】第４の発明では、第１の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得し、更にこの識別子列を識別子列保持手段に格納する。第２の識別子列取得手段が、該識別子列保持手段に保持されている識別子列中から、指定された第２の条件に適合する識別子列を取得し、構成子記述列挿入手段が、この識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する。従って、ある文書中の文書部品を抽出し、これを他の文書中に挿入することができる。

【００３４】第５の発明では、第１の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取

得し、更にこの識別子列を識別子列保持手段に格納する。第２の識別子列取得手段が、該識別子列保持手段に保持されている識別子列中から、指定された第２の条件に適合する識別子列を取得し、構成子記述列変更手段が、この識別子列に対応する構成子記述列を、指定された変更指示情報に従って変更する。従って、指定された文書中の文書部品の属性を指定された属性値に変更することができる。

【００３５】第６の発明では、第１の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得し、更にこの識別子列を識別子列保持手段に格納する。第２の識別子列取得手段が、該識別子列保持手段に保持されている識別子列中から、指定された第２の条件に適合する識別子列を取得し、文書名出力手段が、この識別子列を検査し、この結果が指定された第３の条件に適合する場合は、当該識別子列に対応する構成子群を有する前記予め記述された単数又は複数の文書の文書名を出力する。従って、指定された条件に適合する文書の文書名のみを得ることができる。

【００３６】第７の発明では、第１の発明において、構成子記述列挿入手段が、前記構成子記述列取得手段により得られた構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する。従って、指定された文書中から指定された条件に適合する文書部品を検索し、この文書部品を他の文書中に挿入することができる。

【００３７】第８の発明では、第１の発明において、構成子記述列変更手段が、前記構成子記述列取得手段により得られた構成子記述列を、指定された変更指示情報に従って変更する。従って、指定された文書中から指定された条件に適合する文書部品を検索し、この文書部品の属性を指定された属性値に変更することができる。

【００３８】第９の発明では、第１の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得し、更にこの識別子列を識別子列保持手段に格納する。第２の識別子列取得手段が、該識別子列保持手段に保持されている識別子列中から、指定された第２の条件に適合する識別子列を取得し、構成子記述列変更手段が、この識別子列に対応する構成子記述列を指定された変更指示情報に従って変更し、更に構成子記述列挿入手段が、この変更処理後の構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する。従って、指定された文書中から指定された条件に適合する文書部品を検索し、該文書部品の属性を指定された属性値に変更することができる。

【００３９】第１０の発明では、第１の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている１つ以上の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得し、更に、この識別子列を識別子列保持手段に格納する。第２の識別子列取得手段が、該識別子列保持手段に保持されている識別子列から、指定された第２の条件に適合する識別子列を取得し、構成子記述列取得手段が、この識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書の構成子記述列から取得し、構成子記述列変更手段が、この構成子記述列を指定された変更指示情報に従って変更し、更に構成子記述列挿入手段が、この変更処理後の構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する。従って、指定された文書中から指定された条件に適合する文書部品を検索し、この文書文品の属性を指定された属性値に変更し、更に、この文書部品を指定された文書中に挿入することができる。

【００４０】

【実施例】以下、本発明の第１～第５の実施例を添付図面を参照して説明する。

【００４１】第１の実施例を図１乃至図１１を参照して説明する。

【００４２】図１は本発明に係る文書処理装置の第１の実施例を機能ブロックで示したものである。

【００４３】同図において、文書処理装置は、構成子記述列保持部１１０と、識別子列保持部１２０と、第１の識別子列取得部１３０と、第２の識別子列取得部１４０と、構成子記述列取得部１５０とを有して構成されている。

【００４４】構成子記述列保持部１１０には構造化文書である電子文書が単数又は複数格納されている。この文書の論理構造はグラフ理論におけるグラフによって表現することができるので、この実施例においては、グラフのノードを構成子と定義し、また構成子に対応する記述を構成子記述と定義し、更に構成子を複数並べたものを構成子記述列と定義すると、文書全体は、グラフの全てのノードをそれぞれ表す構成子記述列によって表現される。テキストや図形など、文書の内容を持つ構成子を特に内容部と呼ぶ。また、文書の部品（例えば、章、見出し、図など）も、それを構成するノードを表す構成子記述列によって表現される。

【００４５】ここで、論理構造の構成子記述列の例を以下に示す。

```
logical-object {
  object-type document-logical-root ,
  description-body {
    object-identifier  "3",
    suordinates  "2",
    user-visible-name  "文書論理根" } } ,
logical-object {
  object-type composite-logical ,
  description-body {
    object-identifier  "3 0",
    suordinates  "2",
    user-visible-name  "第１章" } } ,
logical-object {
  object-type basic-logical ,
  description-body {
    object-identifier  "3 0 0",
    user-visible-name  "1.1 節" } } ,
logical-object {
  object-type basic-logical ,
  description-body {
    object-identifier  "3 0 1",
    user-visible-name  "1.2 節" } } ,
logical-object {
  object-type basic-logical ,
  description-body {
    object-identifier  "3 0 2",
    user-visible-name  "1.2 節" } }
```

ここで、例えば

```
logical-object {
object-type document-logical-root ,
description-body {
  object-identifier  "3",
  suordinates  "2",
  user-visible-name  "文書論理根" } } ,
```

で記述される様に「logical-object {　　} 」が構成子であり、「object-identifier "3"」が識別子である。

【００４６】なお上記の例では、５つの構成子で意味のある１つの構成子記述列が表現されている。また上記の構成子記述列から、ある条件に従って、例えば任意の２つの構成子が抽出された場合は、その２つの構成子で意味のある１つの構成子記述列が表現されることとなる。

【００４７】第１の識別子列取得部１３０では、構成子レゾルバ１３１が、予め設定された第１の条件としての条件情報１３２に基づいて、構成子記述列保持部１１０内の構成子記述列群から、その条件に適合する構成子に対応する識別子列を取得するものである。なお条件情報１３２は構成子の検索条件を示す情報である。

【００４８】識別子列保持部１２０は、第１の識別子列取得部１３０により得られた識別子列を保持する。

【００４９】第２の識別子列取得部１４０では、識別子オペレータ１４１が、予め設定された第２の条件としての条件情報１４２に基づいて、識別子列保持部１２０に保持されている識別子列から、その条件に適合する識別子列を取得する。なお条件情報１４２は１つ又は複数の識別子を得るための条件を示す情報である。

【００５０】識別子オペレータ１４１によって取得された識別子列は識別子列保持部１２０に保持される。なおこのとき、識別子列保持部１２０には、第１の識別子列取得部１３０によって取得された識別子列と、第２の識別子列取得部１４０によって取得された識別子列とは、区別されるように保持される。

【００５１】構成子記述列取得部１５０では、構成子アクセサ１５１が、構成子記述の取得条件を示す条件情報１５２に基づいて、構成子記述列保持部１１０内の構成子記述列群から該当する構成子記述列を取得する。ここでは、構成子アクセサ１５１が、第２の識別子列取得部１４０により得られた識別子列を識別子列保持部１２０から得ると共に、この得た識別子列に対応する構成子記述列を、構成子記述列保持部１１０内の構成子記述列群から取得するようになっている。こうして取得された構成子記述列は、既に保持されている他の構成子記述列と区別できるように構成子記述列保持部１１０に保持され、さらに処理操作の対象にもなり得るものである。

【００５２】なおこの実施例においては、第１の識別子列取得部１３０、第２の識別子列取得部１４０、構成子記述列取得部１５０は、それぞれの構成要素の機能を遂行させるためのソフトウェア（プログラム）をプロセッサや中央処理装置などの制御手段が実行することにより実現されている。ここでは、前記各構成要素をそれぞれ独立したプログラム（コマンド）として実現し、構成子記述列、識別子列を表すデータ構造を定め、コマンド間で授受するようにしている。勿論、これを、１つのコマンドに２つ以上の手段に相当する機能を持たせるようにしても良い。この場合は、そのコマンドの入出力のインタフェースは各手段を独立なコマンドとした場合のインタフェースに準ずるものとする。なおこの実施例で使用したコマンドの一例については後述する。

【００５３】上記各コマンドは、ＵＮＩＸの汎用オペレーティングシステムで動作するようになっている。また、各コマンドは単純な機能のみを有するもので、実際に文書処理を行うときには各コマンドが組合わされて使用される。このとき、シェルと呼ばれるコマンドプロセッサ（コマンドインタプリタ）の制御機能を使用して、シェルスクリプトと呼ばれるコマンドプロシジャを作成することにより、高度な処理を実現することができる。あるコマンドの出力を他のコマンド入力とするには、パイプを使用したり、入出力切り替え（リダイレクション）により一時ファイルを生成して使用することができる。

【００５４】例えば、パーソナルメディアから１９８７

年に発行された「落水浩一郎、大木敦男　共訳、UNIX C SHELL　フィールドガイド」（原本：「Gail Anderson andPaul Anderson.　The UNIX C Shell Field Guide).の文献を参照することにより、シェルを実現することができる。

【００５５】構成子記述列保持部１１０、識別子列保持部１２０は、ファイルシステムとパイプ（キュー）を用いて実現している。従ってこれらの構成要素は、格納時に内容と名前を関連付ける機能と、名前で内容をアクセスする機能とを持っており、また内容が一度だけ用いられる場合は、パイプを用いて名前を使わずに一時的に保持する機能を持っている。

【００５６】図２は、第１に示した実施例の装置を実現するためのハードウェア構成を示したものであり、例えば、ワークステーションやコンピュータなどの装置のブロック図を示している。

【００５７】同図において、装置は、入力装置２１０、ディスプレイ２２０、ディスク２３０、主メモリ２４０、中央処理装置（以下、ＣＰＵという）２５０とがバス２６０を介してそれぞれ接続されている。

【００５８】入力装置２１０は、キーボード及びマウスから構成されており、コマンドプロシジャなどの各種のデータの入力や、各種の指令を与えるものである。

【００５９】ディスプレイ２２０は、文書の論理構造、構成子記述列、文書内容、コマンドプロシジャを表示するものである。

【００６０】ディスク２３０には、電子文書（ファイル）つまり文書の論理構造や構成子記述列や文書内容、この実施例の文書処理機能を有するコマンドインタプリタ（プログラム）、コマンドプロシジャ、各種データなどが格納されている。

【００６１】主メモリ２４０には、ディスク２３０からロードされた電子文書ファイル、コマンドインタプリタ、コマンドプロシジャ、各種データ、及び入力装置２１０によって入力されたコマンドプロシジャや各種データや指令を記憶する。なお、この主メモリ２４０には、ＣＰＵ２５０がデータに対する操作を行うための作業領域が設けられており、この作業領域中に、キュー（パイプ）が作成される。また主メモリ２４０に示されている各データやプログラムの配置は、説明用のために配置したものであり、これらは、図示されているデータ構造で配置されるものではない。

【００６２】ＣＰＵ２５０は、コマンドインタプリタをディスク２３０から主メモリ２４０にロードして実行することにより、コマンドプロシジャに対する処理を行う。すなわち、電子文書あるいは文書部品の検索、挿入、合成、更新などの文書処理を行う。またＣＰＵ２５０は、バス２６０を介してこれに接続された各構成要素を制御する。

【００６３】コマンドの説明は、原則的にＵＮＩＸのオンラインマニュアルの形式に従っている。従って、大括弧（［　］）で囲まれたものは、それが省略可能であることを示す。ここでは、縦棒（｜）と中括弧（｛　｝）を導入している。縦棒は、その両方のうちいずれかを選択することを示し、中括弧は、その中の１つが必ず選ばれることを示す。

【００６４】次に、図１に示した機能ブロック図の構成要素と図２に示したブロック図との対応関係について説明する。

【００６５】図１に示した構成子記述列保持部１１０及び識別子保持部１２０はディスク２３０、主メモリ２４０に対応し、第１の識別子列取得部１３０、第２の識別子列取得部１４０、構成子記述列取得部１５０はＣＰＵ２５０（実際にはＣＰＵ２５０がコマンドインタプリタを実行して、コマンドを解釈することにより実現される）に対応している。

【００６６】なお後述する第２～第５の実施例の装置も図２に示したハードウェア構成で実現されている。

【００６７】次に、この実施例で使用するコマンドについて説明する。ここでは、以下のようにコマンド群のカテゴリ（範疇）を定義する。

［カテゴリ　　：機能説明］
◇構成子レゾルバ　：構成子記述列から条件を満たす識別子列を得る。
◇識別子オペレータ：識別子列（の列）から条件を満たす識別子列を得る。
◇構成子アクセサ　：識別子列と文書から構成子記述列を得る。
◇部品挿入　　：文書に部品を挿入する。
◇属性更新　　：指定された構成子記述の属性値を更新する。
◇文書オペレータ　：識別子列と文書名から条件を満たす文書名を得る。
◇文書レゾルバ　：文書名の列から条件を満たす文書名の列を得る。
◇文書アクセサ　：文書名から文書の内容を得る。

なお、図１に示した構成子レゾルバ１３１は上記コマンドの構成子レゾルバに相当し、図１に示した識別子オペレータ１４１は上記コマンドの識別子オペレータに相当し、図１に示した構成子アクセサ１５１は上記コマンドの構成子アクセサに相当する。

【００６８】次に、カテゴリ毎に、カテゴリに属するコマンドの名前、形式、機能、オプションについて説明する。

【００６９】◇構成子レゾルバ
構成子記述列と条件を入力として、条件を満たす構成子の識別子列を得るコマンドである。条件としては、属性と指定された数値との比較（＝、≠、＜、＞、≦、≧）、属性値と指定された文字列との辞書順比較や照合などがあげられる。このカテゴリに属するコマンドの例

を２つ挙げる。

【００７０】［名前］select-content　：内容部を扱う構成子レゾルバ

［形式］select-content string ［filename］

［機能説明］filename（ファイル名）で指定された文書ファイルから、stringで与えられた文字列が部分文字列として現れる構成子（内容部）を求める。大文字・小文字は区別されない（例えばstringとして“ａｂｃ”が指定されたとき、“ＡＢＣ”という文字列とも一致する。）。filenameが与えられなかったときには、標準入力（例えば端末。以下同様とする）から構成子記述列を読み込む。

【００７１】［名前］select-type　：属性“uobject-type”を扱う構成子レゾルバ

［形式］select-type ［-e］string［filename］

［機能説明］filename（ファイル名）で指定された文書ファイルから、属性“uobject-type”の値が与えられた条件に適合する構成子の識別子を求め、標準出力（例えば端末。以下同様とする）に書き出す。filenameが与えられなかったときには、標準入力から構成子記述列を読み込む。-eオプションが指定されない場合は、属性“uobject-type”の値がstringで与えられた文字列と完全に一致するものを求める。なお大文字・小文字は区別されない。

［オプション］－e　：stringが属性“uobject-type”の値の部分文字列として出現する構成子を求める。

【００７２】◇識別子オペレータ

識別子列と条件を入力として、指定された属性の値が条件を満たすような構成子の識別子を列挙するコマンドである。

【００７３】このカテゴリに属するコマンドの例を、４つ挙げる。

【００７４】［名前］ａｂｏｖｅ

［形式］above ［filename］

［機能説明］filename（ファイル名）で指定された２つの識別子列を読み、最初の識別子列に含まれる識別子のうち、２番目の識別子列に含まれるいずれかの識別子の祖先になっている識別子を選択し、標準出力に書き出す。filenameが省略された場合には、標準入力から識別子列を読み込む。

【００７５】［名前］ｂｅｌｏｗ

［形式］below ［filename］

［機能説明］filename（ファイル名）で指定された２つの識別子列を読み、最初の識別子列に含まれる識別子のうち、２番目の識別子列に含まれるいずれかの識別子の子孫になっている識別子を選択し、標準出力に書き出す。filenameが省略された場合には、標準入力から識別子列を読み込む。

【００７６】［名前］ｂｏｔｔｏｍ

［形式］bottom［filename］

［機能説明］filename（ファイル名）で指定された識別子列ファイルから、他の識別子の先祖になっていない識別子だけを選び出して標準出力に書き出す。filenameが指定されなければ、標準入力から識別子列を読み込む。例えば入力として、＜３１＞　＜３１２＞　＜３１２３＞　＜３１３＞　＜３３３＞が与えられると（上記例では５つの識別子からなる１つの識別子列）、出力として、＜３１２３＞　＜３１３＞　＜３３３＞が得られる（１つの識別子列）。

【００７７】［名前］ｕｎｉｏｎ

［形式］union ［filename］

［機能説明］filename（ファイル名）で指定されたファイルに含まれる２つの識別子列から和集合を求め、標準出力する。filenameが指定されなければ、標準入力から識別子列を読み込む。例えば入力として、＜３１＞　＜３１２３＞　＜３３３＞と＜３１２＞　＜３１２３＞　＜３３３＞の２つの識別子列が与えられると（上記例では、３つの識別子列からなる１つの識別子列と、３つの識別子列からなる１つの識別子列とからなる複数の識別子列）、出力として、＜３１＞　＜３１２＞　＜３１２３＞　＜３３３＞が得られる（１つの識別子列）。

【００７８】なおこのｕｎｉｏｎコマンドの様に、２つの識別子列（集合）を入力として、集合演算を行う他の例としては、次のようなコマンドがある。

（ａ）intersection：２つの識別子列（集合）を入力として共通集合を求める。

（ｂ）differrence　：２つの識別子列（集合）を入力として差集合を求める。

（ｃ）unique：識別子列を入力として、識別子の集合を求める（重複しないようにする）。

などがある。

【００７９】◇構成子アクセサ

構成子記述列と識別子と条件を入力として、条件を満たす構成子記述列を得るコマンドである。条件として、識別子オペレータの項で説明した接続情報を用いることができる。条件が与えられなければ、識別子列中の各識別子に対応する構成子の構成子記述を取り出す。

【００８０】このカテゴリのコマンドの例を１つ挙げる。

【００８１】［名前］ｅｘｔｒａｃｔ

［形式］extract ［-s］id-file filename

［機能説明］id-file （識別子列ファイル）で指定された文書ファイルから、id-file で指定された識別子列に対応する構成子記述を求める。id-file として“－”が与えられたときには、標準入力から識別子を読み込む。filename（ファイル名）として“－”が与えられたときには、標準入力から構成子記述列を読み込む。id-fileとfilenameの両方に“－”を与えることはできない。

［オプション］－s　：指定された識別子を持つ構成子を頂点とする部分木の構成子記述列を取り出す。

【００８２】◇部品挿入
挿入箇所を示す識別子と、文書部品ファイル（構成子記述列）と、文書ファイル（構成子記述列）とを入力して、文書ファイルに部品を挿入するコマンドである。部品を挿入したときには、識別子が文書内で一意になるように、必要に応じて識別子の再割り当てを行う。

【００８３】このカテゴリに属するコマンドの例を１つ挙げる。

【００８４】［名前］ｉｎｓｅｒｔ
［形式］

insert {-ec | -yc | -eb | -yb} [-q] {-p id | id-filename}
part-filename doc-filename

［機能説明］構成子記述列を、文書中の指定された挿入位置に挿入して、標準出力に書き出す。構成子記述列と文書ファイルはそれぞれ、part-filename 、doc-filenameで指定したファイルから読み込まれる。挿入位置は、識別子ファイルid-filename の先頭か、－ｐオプションで指定された識別子を基準に、他のオプションによって決定される。part-filename 、doc-filename、id-filename のうちいずれかに“－”を指定したときは、それを標準入力から読み込む。“－”を複数回指定することはできない。出力中の構成子記述の識別子は、Ｄｅｗｅｙの１０進分類法（Donald E. Knuth, “The Art of Programming”, Addison-Wesley Publishing Company, 1973。邦訳：筧捷彦・米田信夫共訳、“基本算法／情報構造”，サイエンス社，１９８７年）に従うよう振り直される。
［オプション］
－ec　：指定された識別子を持つ構成子の最初の子供になるように挿入する（eldest child）。
－yc　：指定された識別子を持つ構成子の最後の子供になるように挿入する（youngest child）。
－eb　：指定された識別子を持つ構成子の兄になるよに挿入する（elder brother ）。
－yb　：指定された識別子を持つ構成子の弟になるように挿入する（younger brother ）。
－p　id　：構成子記述列を挿入するノードの識別子を指定する。例えば、－ｐ“３０１”のように指定する。

【００８５】◇属性更新

id-doc {-gt | -lt | -ge | -le | -eq | -ne} num {id-file | -} doc-file

［機能説明］id-file （識別子列ファイル）で指定されたファイルに含まれる識別子の数が、与えられた条件を満たすときdoc-file（文書ファイル）を出力する。id-file の代わりに“－”が与えられたときは、標準入力から識別子列を読み込む。
［オプション］
－gt　num　：id-file に含まれる識別子の数がnum より大きいときに真。
－lt　num　：id-file に含まれる識別子の数がnum より小さいときに真。
－ge　num　：id-file に含まれる識別子の数がnum 以上のときに真。
－le　num　：id-file に含まれる識別子の数がnum 以下のときに真。
－eq　num　：id-file に含まれる識別子の数がnum と等しいときに真。
－ne　num　：id-file に含まれる識別子の数がnum と等しくないときに真。

【００８６】このカテゴリに属するコマンドの例を１つ挙げる。

【００８７】［名前］ｃｈｖａｌ
［形式］

chval {-n | -t} old new {id-file | -} doc-file

［機能説明］doc-file（文書ファイル）中の構成子記述列のうち、id-file （識別子列ファイル）の識別子列に対応する構成子の属性値に含まれる文字列old を文字列new で置換する。id-file の代わりに、“－”が与えられたときには、標準入力から識別子列を読み込む。id-file に識別子が全く含まれていないときには、doc-file のすべての構成子を操作対象とする。
［オプション］
－n　：属性 "user visible name"を操作対象とする。
－t　：属性 "uobject type" を操作対象とする。

【００８８】◇文書オペレータ
識別子列と文書名と条件を入力とし、文書名を得るオペレータである。与えられた条件が満たされればその文書名を出力し、そうでなければ何も出力しない。このカテゴリに属するコマンドの例を１つ挙げる。

【００８９】［名前］ｉｄ－ｄｏｃ
［形式］

［機能説明］構成子記述列と識別子列と変更指示を入力として、指示に従って属性の更新を行うコマンドである。識別子が与えられない場合には、全構成子記述の属性の更新を行う。

【００９０】◇文書レゾルバ
文書名列と条件を入力として、条件を満たす文書名列を得るコマンドである。このカテゴリに属するコマンドは、指定された属性の値が条件を満たすような文書の名前を列挙する。条件としては、属性と指定された数値の比較（＝、≠、＜、＞、≦、≧）、属性値と指定された文字列の辞書順比較や照合などがあげられるる。例えば、文書名、著者、キーワードに特定の文字列を含む文書の列挙、ページ数が指定された値以上（あるいは以下）の文書の列挙、を行うコマンドがある。

【００９１】◇文書アクセサ
文書名を引数として文書の内容である構成子記述列を得

るコマンドである。例えば、指定された文書ファイルの内容を標準出力に書き出すコマンドがある。なおこのカテゴリのコマンドの機能は、ファイルシステムに持たせても良い。

【００９２】ところで、接続関係に基づく識別子オペレータの例としては、指定された識別子に対応する構成子との距離が指定された値以下である構成子の識別子を列挙すコマンドがある。例えば、図３に示すようなネットワーク構造において、ノード１との距離が１以下のノードは｛１，２｝である。なお４つの図形“○”はノードを表している。

【００９３】接続関係に基づく演算を行うコマンドの実現方法について述べる。

【００９４】簡単のため、入力となる各構成子記述に対しては、構成子に文書内で一意な識別子があらかじめ付与されているものとする。

【００９５】データ構造としては、識別子を添数とする隣接行列を用いる。識別子は自然数とあるとし、論理構造がなすグラフがストリクトであるとすれば、ｎ個の構成子記述に対して、隣接行列として以下の様なｎ次の正方行列$A=a_{ij}$（$1 \leqq i, j \leqq n$）を用意することにより、隣接情報を得ることができる。

【００９６】ここにストリクトとはループと多重弧が存在しないこと。ここで、ループとはあるノードからそれ自身へのアークであり、多重弧とはあるノードから他の特定のノードに至る複数のアークである。

【００９７】

$$a_{ij}=\begin{cases} 1 & \text{アーク}(i, j)\text{が存在する} \\ 0 & \text{アーク}(i, j)\text{が存在しない} \end{cases}$$

例えば、A^1（＝A）、A^2（＝A・A）、…A^nから、それぞれあるノードから１、２、…、ｎ回リンクを辿って到達できるノードを求めることができる。

また、$\sum_{i=1}^{n} A^i$ から

リンクをｎ以下の回数だけ辿ったときに到達可能なノードを求めることができる。

$$A^2 = A \cdot A = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$A^3 = A^2 \cdot A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$A^4 = A^3 \cdot A = 0$$

である。

【００９９】Ａで定義されるグラフは、ＤＡＧ（directed acyclic graph）であるので、

【００９８】ところで、図３に示すようなネットワークであれば、隣接行列Ａは、

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

となる。このとき、

$C=\sum_{i=1}^{\infty} A^i$ とすれば、

$$C = \sum_{i=1}^{3} A^i = \begin{bmatrix} 0 & 1 & 2 & 1 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

となる。

【０１００】$C=c_{ij}$（$0 \leqq i, j \leqq n$）の成分ｃ

$_{ij}$は、ノードｉからノードｊに至る経路の総数を表す。

【０１０１】このような接続情報に関する性質を用いて、識別子と条件を入力として、識別子列を出力するコマンドを実現することができる。

【０１０２】一般に構造化文書の論理構造は木構造で表されるが、木構造は接続に制限を設けた特殊なネットワーク構造であるので、上述した隣接行列を用いて親子関係などの子孫関係を求めることができる。従って、本実施例により、構造化文書もネットワーク構造で表現される文書と全く同様に操作することができる。

【０１０３】構成子記述がテキストで表現されている場合には、本実施例に組み合わせて、ＵＮＩＸの標準コマンドｓｅｄ、ａｗｋといった汎用のテキスト操作ツール（あるいは言語）を用いることができる。

【０１０４】オブジェクト指向データベースでは、オブジェクトに対して内部的に一意な識別子を割り当てている。従って、オブジェクト指向データベースにおいても、識別子の内部表現に自然数が１対１に対応するようにすれば、隣接行列を用いることにより、識別子と条件を入力して識別子列を出力するようなプログラムを作成することができる。

【０１０５】次に、図４に示すような木構造によって表現される文書を処理対象とする場合について説明する。

【０１０６】ここでは、木構造の各ノードにアドレスを割り当て、アドレスの集合に対する演算を定義し、その演算を行うコマンドの実現方法を説明する。また、構成子アクセサでの処理において、アドレスの集合に対する演算に基づいてアクセスする方法も説明する。

【０１０７】図５に示すように、図４に示す木構造の各ノードにＤｅｗｅｙの１０進分類法によってアドレスを割り当てる。アドレスは数値の並びである。

【０１０８】このようにして与えられたアドレスから、ただちに接続関係を得ることができる。アドレスａｄｄｒの長さをlength（addr）、アドレスａｄｄｒの先頭のｌｅｎ桁をｈｄ（ａｄｄｒ，ｌｅｎ）で表すことにすると、アドレスａに対応するオブジェクトがアドレスｂに対応するオブジェクトの先祖であるための必要十分条件は、ａ＝ｈｄ（ｂ，length（ａ））である。

【０１０９】例えば、ａ＝〈１２〉，ｂ＝〈１２１２〉とすれば、

$$hd(b, length(a)) = hd(\langle 1212\rangle, 2) = \langle 12\rangle = a$$

である。

【０１１０】２つのアドレスを引数とする関数ｌｕｂを次のように定義する。

$$lub(a, b) = hd(a, \max\{i \mid hd(a, i) = hd(b, i)\})$$

ｌｕｂ（ａ，ｂ）はアドレスａ，ｂに対応するオブジェクトの共通の祖先のうち、もっとも若いもののアドレスである。

【０１１１】この関数は図６に示すような処理手順により、プログラムとして実現することができる。

【０１１２】第２の識別子列取得部は、アドレスａ、ｂを入力し（ステップ６０１）、ｉを値「１」と定義する（ステップ６０２）。次に、ａ（ｉ）＝ｂ（ｉ）の式が成立するか否かを判断し（ステップ６０３）、成立する場合は、ａ（ｉ）をｐｒｉｎｔ（つまり出力）し（ステップ６０４）、ｉをｉ＋１と再定義する（ステップ６０５）。その後、ａ（ｉ）＝ｂ（ｉ）の式が成立するか否かを判断し（ステップ６０６）、成立する場合は上記ステップ６０４に戻りこのステップ以降を実行し、不成立の場合は、リターンする。ステップ６０３で式が不成立の場合はリターンする。

【０１１３】次に関数ｌｕｂを用いて、述語ａｎｃ、ｄｅｓｃを定義する。

$$anc(a, b) \Leftrightarrow lub(a, b) = a$$

$$desc(a, b) \Leftrightarrow lub(a, b) = b$$

すなわち、ａｎｃ（ａ，ｂ）はａがｂの祖先であるか等しいとき、かつそのときに限り真、ｄｅｓｃ（ａ，ｂ）はａがｂの子孫であるか等しいとき、かつそのときに限り真である。

【０１１４】また述語ａｎｃ、ｄｅｓｃを用いて、集合演算ａｂｏｖｅ、ｂｅｌｌｏｗを定義する。

$$above(A, B) = \{a \in A \mid (\exists b \in B)\ (anc(a, b))\}$$

$$below(A, B) = \{a \in A \mid (\exists b \in B)\ (desc(a, b))\}$$

ここで、Ａ，Ｂは集合である。

ａｂｏｖｅ（Ａ，Ｂ）を求めることは、Ａの要素のうちＢのどれかの祖先であるか等しいものを選択する操作に等しい。ｂｅｌｏｗ（Ａ，Ｂ）を求めることは、Ａの要素のうちＢのどれかの子孫であるか等しいものを選択する操作に等しい。

【０１１５】更に、述語ａｎｃ、ｄｅｓｃを用いて、集合演算ｔｏｐ、bottomを定義する。

$$top(A)=\{a\in A\mid(\forall b\in A)(a=b\vee\neg anc(b,a))\}$$
$$bottom(A)=\{a\in A\mid(\forall b\in A)(a=b\vee\neg desc(b,a))\}$$

ｔｏｐ（Ａ）を求めることは、Ａの要素のうち他の要素の子孫になっていないものだけを選択する操作に等しい。bottom（Ａ）を求めることは、Ａの要素のうち他の要素の先祖になっていないものだけを選択する操作に等しい。

【０１１６】同様にして述語mother、daughterを定義し、これらを用いて集合演算parent、child を定義できる。

$$mother(a,b)\Leftrightarrow above(a,b)\wedge length(a)=length(b)-1$$
$$daughter(a,b)\Leftrightarrow below(a,b)\wedge length(a)+1=length(b)$$
$$parent(A,B)=\{a\in A\mid(\exists b\in B)(mother(a,b))\}$$
$$child(A,B)=\{a\in A\mid(\exists b\in B)(daughter(a,b))\}$$

ここで、Ａ，Ｂは集合である。

parent（Ａ，Ｂ）を求めることは、Ａの要素のうちＢのどれかの親であるか等しいものを選択する操作に等しい。child （Ａ，Ｂ）を求めることは、Ａの要素のうちＢのどれかの子であるか等しいものを選択する操作に等しい。

【０１１７】図４に示す木構造に、行きがけ順（preorder）に番号を振ったものを図７に示す。構造化文書の論理構造においては、ほとんどの場合、行きがけ順に従ってレイアウトされるので、この順番は特別な意味を持つ。例えば図４の論理構造は、通常、第１章、１．１節、１．２節、１．３節、第２章、２．１節、２．２節、２．２．１節、２．２．２節、２．２．３節、の順にレイアウトされる。

【０１１８】図４に示す木構造の各ノードのアドレスを行きかけ順に並べると、〈１〉、〈１１〉、〈１１１〉、〈１１２〉、〈１１３〉、〈１２〉、〈１２１〉、〈１２２〉、〈１２２１〉、〈１２２２〉、〈１２３〉となる。

【０１１９】２つのアドレスのうち行きがけ順でどちらが先にあるかをアドレスが持つ情報だけから判定することができる。アドレスａｄｄｒのｉ番目の要素をａｄｄｒ$_i$で表すことにする。ｉ＞ｌｅｎ（ａｄｄｒ）のときには、ａｄｄｒ$_i$＝０と定める。

【０１２０】ここで述語ｐｒｅｄを次のように定義する。

$$pred(a,b)\Leftrightarrow a_{lub(a,b)+1}\leq b_{lub(a,b)+1}$$

ｐｒｅｄ（ａ，ｂ）は、ａ、ｂのうち行きがけ順で先に表されるものを求める関数である。

【０１２１】同様に、述語ｓｕｃｃを次のように定義する。

$$succ(a,b)\Leftrightarrow a_{lub(a,b)+1}\geq b_{lub(a,b)+1}$$

ｓｕｃｃ（ａ，ｂ）はａ，ｂのうち行きがけ順で後に表されるものを求める関数である。

【０１２２】そして述語ｐｒｅｄ、ｓｕｃｃを用いて、集合演算ｈｅａｄ、ｔａｉｌを定義する。

$$head(A,B)=\{a\in A\mid(\exists b\in B)(pred(a,b))\}$$
$$tail(A,B)=\{a\in A\mid(\exists b\in B)(succ(a,b))\}$$

ここで、Ａ，Ｂは集合である。

ｈｅａｄ（Ａ，Ｂ）を求めることは、Ａの要素のうちＢのどれかに等しいか、行きがけ順でそれよりも早く出現するものを選択する操作にひとしい。ｔａｉｌ（Ａ，Ｂ）を求めることは、Ａの要素のうちＢのどれかに等しいか、行きがけ順でそれよりも後に出現するものを選択する操作に等しい。

【０１２３】上述の様に、接続関係（木構造における子孫関係、行きがけ順での順序関係）に基づく演算が定義でき、それに従って処理を行う識別子オペレータを実現することができる。

【０１２４】いま、Ｔを与えられた木構造に属する構成子の識別子の全体として、

Ｔ＼ｂｅｌｏｗ（Ｔ，Ａ）はＡの各要素を頂点とする部分木を、もとの木構造から除去する操作に等しい。このような操作を行う構成子アクセサを実現することができる。

Ａ⊂Ｔとすると、

【０１２５】次に第１の実施例の処理動作について、図８乃至図１１を参照して説明する。なお、図８は全体の処理の流れを示すメインルーチンを示し、図９はその処理において実行されるスクリプトの実行処理を示すサブルーチンを示し、図１０はそのスクリプトの実行処理において実行される基本単位の処理を示すサブルーチンを示し、図１１はその実行単位の処理において実行される基本実行単位の処理を示すサブルーチンを示している。

【０１２６】ここで、実行単位と基本実行単位を次の様に定義する。

【０１２７】ａ，ｂ，ｃ、…，ｚをコマンドとする。単

独のコマンドは基本実行単位である。コマンドとその引数も基本実行単位である。コマンド（とその引数）の入力、出力の一方あるいは両方をリダイレクト（切り替え）したものも基本実行単位である。基本実行単位は実行単位である。コマンドをセミコロンで繋いもの、例えば、

ａ；ｂ

は個々のコマンドがそれぞれ１つの基本実行単位である。つまり、上記の例では、ａ，ｂの２つの基本実行単位がある。コマンドをパイプでつないだもの、例えば、

ａ｜ｂ

は、全体で１つの実行単位である。実行単位を対になった丸括弧でくくったもの、例えば、

（ａ；ｂ）

は、１つの基本実行単位である。より複雑な例として、

（ａ；ｂ）｜ｃ

を考えると、これ全体で１つの実行単位である。

【０１２８】さて全体の処理の流れについて説明する。図８に示すように、シェル（コマンドインタプリタ）は、スクリプトの実行か否かを判断し（ステップ８０１）、スクリプトの実行の場合には、そのスクリプトを実行する（ステップ８０２）。

【０１２９】一方、ステップ８０１においてスクリプトの実行でない場合は、スクリプトを実行単位に分割し（ステップ８０３）、未実行の実行単位が存在するか否かを判断する（ステップ８０４）。ここで、未実行の実行単位が存在する場合は、最も左の実行単位を取り出し（ステップ８０５）、この実行単位を処理し（ステップ８０６）、その後、未実行の実行単位が存在するか否かを判断する（ステップ８０７）。ここで、未実行の実行単位が存在する場合は、上記ステップ８０５に戻りこのステップ以降を実行する。

【０１３０】なお、ステップ８０４、８０７において、未実行の実行単位が存在しない場合はリターンする。

【０１３１】次に、スクリプトの実行処理について説明する。

【０１３２】図９に示すように、シェルは、ファイルの終りであるかを判断し（ステップ９０１）、ファイルの終りでない場合はバッファをクリアし（ステップ９０２）、スクリプト中の１行を読み込んで、バッファに追加（格納）すると共に（ステップ９０３）、その読み込んだ行の最後のトークンが“＼”であるか否かを判断する（ステップ９０４）。

【０１３３】ステップ９０４において、“＼”の場合は上記ステップ９０３に戻りこのステップ以降を実行し、反対に、“＼”でない場合は変数を展開すると共に（ステップ９０５）、スクリプトを実行単位に分割し（ステップ９０６）、更に未実行の実行単位は存在するか否かを判断する（ステップ９０７）。

【０１３４】ステップ９０７において、未実行の実行単位が存在する場合は、最も左の実行単位を取り出し（ステップ９０８）、該実行単位を処理し（ステップ９０９）、未実行の実行単位はあるか否かを判断する（ステップ９１０）。

【０１３５】ステップ９１０において、未実行の実行単位が存在する場合は上記ステップ９０８に戻りこのステップ以降を実行し、反対に未実行の実行単位が存在しない場合はファイルの終りであるかを判断する（ステップ９１１）。

【０１３６】なおステップ９０７において未実行の実行単位が存在しない場合はステップ９１１に進む。

【０１３７】ステップ９１１において、ファイルの終りでない場合は上記ステップ９０２に戻りこのステップ以降を実行する。ステップ９１１、ステップ９０１においてファイルの終りの場合は処理を終了し、図８のステップ８０２にリターンする。

【０１３８】次に、実行単位の処理について説明する。

【０１３９】図１０に示すように、シェルは、実行単位を基本実行単位に分割し（ステップ１００１）、パイプ、リダイレクションの処理を行う（ステップ１００２）。

【０１４０】次に未実行の基本実行単位が存在するか否かを判断し（ステップ１００３）、未実行の基本実行単位がある場合は、最も左の基本実行単位を取り出し（ステップ１００４）、この基本実行単位の処理を実行し（ステップ１００５）、その後、未実行の基本実行単位が存在する否かを判断する（ステップ１００６）。

【０１４１】ステップ１００６において、未実行の基本実行単位が存在する場合は、上記ステップ１００４に戻りこのステップ以降を実行する。なおステップ１００６及びステップ１００３において、未実行の基本実行単位が存在しない場合は、処理を終了し、リターンする。ここで、この実行単位の処理が、スクリプトの実行処理のサブルーチンとして実行されたときは図８のステップ８０２にリターンし、コマンドの実行処理のサブルーチンとして実行されたときは図９のステップ９０９にリターンする。

【０１４２】次に、基本実行単位の処理について説明する。

【０１４３】図１１に示すように、シェルは、該当する基本実行単位における最も左のトークンが“（”であるか否かを判断し（ステップ１１０１）、“（”である場合は基本実行単位から左右の丸括弧を除去した文字列を引数として、サブシェルを起動し（ステップ１１０２）、一方、“（”でない場合はコマンドを起動する（ステップ１１０３）。ステップ１１０２において、サブシェルが起動されると、このサブシェルによって、図８～図１１に示される処理手順が実行される（つまり図８～図１１に示される処理が再帰的に実行される）。

【０１４４】ステップ１１０２、１１０３が終了した場

合は、図１０のステップ１００５にリターンする。

【０１４５】なお、スクリプトの実行には既存のシェルを用いることができる。

【０１４６】ここで、図１２、図１３に示される様な、処理対象となる文書を用いて文書処理を説明する。

【０１４７】なお図１２は図の論理構造を示したものであり、図１３は章の論理構造を示したものであり、これら各図において、矩形内に示されている文字列が属性“uobject-type”の値として構成子に指定されているものとする。

【０１４８】また図１２に示した論理構造では、図は、グラフィックスとオプショナルな図見出しからなっている。“Figure”と記述された構成子が、図を表す論理構造の根である。また"Raster Graphics”と記述された構成子がグラフィックスを含む内容部である。また“Caption Number”と記述された構成子が、図見出し番号を持つ内容部である。更に“Caption Number”と記述された構成子が、図見出しのテキストを持つ内容部である。

【０１４９】更に図１３に示した論理構造では、章（或いは節）は、見出し、段落の繰り返し、オプショナルな部分節からなっている。従って、章（あるいは節）は入れ子になって良い。“Numbered Segment”と記述された構成子が、章（節）を表す論理構造の根である。"Title”と記述された構成子が見出しを表す部分木の根である。“Title Number”と記述された構成子が、章（節）見出し番号をもつ内容部である。“Title Text”と記述された構成子が、章（節）見出しのテキストを持つ内容部である。

【０１５０】次に、実際に処理を行うためのコマンドの仕様例、及びそのコマンドを用いたスクリプト（コマンドプロシジャ）の例を以下に示す。すなわちBourneシェルのスクリプトである（Bourneシェルの仕様についてはＵＮＩＸのオンラインマニュアルを参照）。

【０１５１】セミコロンで区切られたコマンドは順次実行される。丸括弧（すなわち“(”や“)”である）で囲まれたコマンドの列はサブシェルで実行され、あたかも単独のコマンドであるかの様に扱われる。行末のバックスラッシュは、次の行が継続行であることを示す。

【０１５２】＄１、＄２などは、スクリプトに与えられた引数を参照するための変数（positional parameter）てある（具体的には文書名つまりファイル名である）。例えばスクリプト名がｆｏｏであるとして、コマンドラインから

％　ｆｏｏ　ｗｈｏ　ａｒｅ　ｙｏｕ

と入力したとすると、＄１はｗｈｏ、＄２はａｒｅ、＄３はｙｏｕである。

【０１５３】◇［スクリプト例１］

```
#!/bin/sh
(select-type -e  Figure $1 ; select-type -e Title $1) | union | \
    extract -s - $1
```

このスクリプトは指定された文書の図と見出しを検索する例である。引数は検索対象となる文書名である。複数の引数が与えられた場合、２つ目以降の引数は無視される。

【０１５４】最初のselect-type コマンドの実行により、指定されたファイル中の構成子記述列のうち、属性“uobject-type”の値が“Figure”を含む構成子（すなわち図を表す部分木の根）の識別子列が求められる。同様に、２番目のselect-type コマンドの実行により、属性“uobject-type”の値が"Title”を含む構成子（すなわち見出しの部分木の根）の識別子列が求められる。union コマンドの実行により、最初のselect-type コマンドの実行と２番目のselect-type コマンドの実行とによって得られた２つの識別子列（集合）の和集合が得られる。最後にextract コマンドの実行により、union コマンドの実行により得られた識別子列に含まれる識別子を持つ構成子を頂点とする部分木の構成子記述列がファイルから抜き出される。

【０１５５】上述したように第１の実施例によれば、指定された構造化文書から条件に適合する文書部品を検索することができる。

【０１５６】次に本発明の第２の実施例を図１４を参照して説明する。

【０１５７】図１４は本発明に係る文書処理装置の第２の実施例を機能ブロック図で示したものである。この機能ブロック図は、図１に示した第１の実施例の機能ブロック図の構成において、構成子記述列取得部１５０を削除し、構成子記述列挿入部１２００を追加した構成になっている。なお同図において、図１に示した構成要素と同様の機能を果たす部分には同一の符号を付している。

【０１５８】構成子記述列挿入部１２００では、部品挿入部１２１０が、第２の識別子列取得部１４０により得られた識別子列に対応する構成子記述列を、指定された条件としての条件情報１２２０に従って、構成子記述列保持部１１０に保持されている構成子記述列群に挿入する。なお条件情報１２２０は文書部品を挿入するための条件を示す情報である。部品挿入部１２１０による挿入処理後の構成子記述列は、既に保持されている他の構成子記述列と区別できるように構成子記述列保持部１１０に保持され、さらに処理操作の対象にもなり得るものである。

【０１５９】この第２の実施例においても、構成子記述列挿入部１２００は、この構成要素の機能を遂行させるためのソフトウェア（プログラム）を制御手段が実行す

ることにより実現される。従って、部品挿入部１２１０は上述した部品挿入のコマンド（つまりｉｎｓｅｒｔコマンド）に相当する。

【０１６０】なおこの第２の実施例では、第２の識別子列取得部１４０の識別子オペレータ１４１は上述したｂｏｔｔｏｍコマンドに相当するように設定されている。またスクリプトを実行するシェルの処理は、上記第１の実施例の図８乃至図１１で説明した処理と同様である。

【０１６１】次に、実際に処理を行うためのコマンドの仕様例、及びそのコマンドを用いたスクリプトの例を以下に示す。

【０１６２】◇［スクリプト例２］

```
#!/bin/sh

select-type -e Segment $2 | bottom > pid
insert -yc pid $1 $2
```

このスクリプトは、部品として挿入されるファイル（以下、部品ファイルという）中の構成子記述子列を他のファイル（以下、文書ファイルという）中の構成子記述列に含まれる最下位の節のうち先頭にあるものの末子になるように挿入される例である。

【０１６３】このスクリプトの第１引数は部品として挿入される部品ファイル、第２引数は部品が挿入される文書ファイルである。３個以上の引数が与えられた場合、３つ目以降の引数は無視される。

【０１６４】select-type コマンドの実行により、文書ファイルの構成子のうち、属性“uobject-type”の値が"Segment" を含む構成子（すなわち節を表す部分木の根）の識別子列が求められる。次に、bottomコマンドの実行により、select-type コマンドの実行により得られた識別子列のうち最下位にある構成子の識別子列だけが得られる。bottomコマンドの実行による出力は、作業用のファイルpid に格納される。最後に、insertコマンドの実行により、部品ファイルの構成子記述列が、文書ファイルに挿入される。挿入位置は、ファイルpid 中の先頭の識別子に対応する構成子の末子である。

【０１６５】上述した様に第２の実施例によれば、ある構造化文書中の文書部品を抽出し、この文書部品を他の構造化文書中に挿入することができる。

【０１６６】次に本発明の第３の実施例を図１５を参照して説明する。

【０１６７】図１５は本発明に係る文書処理装置の第３の実施例を機能ブロック図で示したものである。この機能ブロック図は、図１に示した第１の実施例の機能ブロック図の構成において、構成子記述列取得部１５０を削除し、構成子記述列変更部１３００を追加した構成になっている。なお同図において、図１に示した構成要素と同様の機能を果たす部分には同一の符号を付している。

【０１６８】構成子記述列変更部１３００では、部品変更部１３１０が、第２の識別子列取得部１４０により得られた識別子列に対応する構成子記述列を、文書部品の属性を変更するための条件を示す情報である指定された変更指示情報１３２０に従って変更する。この結果である変更処理後の構成子記述列は、既に保持されている他の構成子記述列と区別できるように構成子記述列保持部１１０に保持され、さらに処理操作の対象にもなり得るものである。

【０１６９】この第３の実施例においても、構成子記述列変更部１３００は、この構成要素の機能を遂行させるためのソフトウェア（プログラム）を制御手段が実行することにより実現される。従って、部品変更部１３１０は上述した部品変更のコマンド（つまりｃｈｖａｌコマンド）に相当する。

【０１７０】なおこの第３の実施例では、第１の識別子列取得部１３０の構成子レゾルバ１４１は上述したｓｅｌｅｃｔ－ｔｙｐｅコマンドに相当し、第２の識別子列取得部１４０の識別子オペレータ１４１は上述したｕｎｉｏｎコマンドに相当するように設定されている。またスクリプトを実行するシェルの処理は、上記第１の実施例の図８乃至図１１で説明した処理と同様である。

【０１７１】次に、実際に処理を行うためのコマンドの仕様例、及びそのコマンドを用いたスクリプトの例を以下に示す。

【０１７２】◇［スクリプト例３］

```
#!/bin/sh
(select-type -e Figure $1 ; select-type -e Segment $1) | union | \
    chval -n Patent 特許 - $1 > $1.$$
mv  $1.$$ $1
```

このスクリプトは、指定された文書の図或いは節を表す部分木の根の属性"user visible name" に現れる文字列“Patent”を“特許”に変更する例である。

【０１７３】変数$$は、このスクリプトが実行されたときの/bin/sh のプロセス番号に展開される。

【０１７４】２つのselect-type コマンドは、第１の実施例のスクリプト例１におけるselect-type コマンドと同様の目的で用いられる。union コマンドの実行により、２つのselect-type コマンドそれぞれの実行結果である識別子列の和集合が求められる。chval コマンドの実行により、union コマンドの実行によって得られた識別子列で指定された構成子記述列の属性"user visible name" に含まれる文字列“Patent”が“特許”に置換される。chval コマンドの実行結果である構成子記述列は、作業用ファイルに格納される。最後に、ＵＮＩＸの標準コマンドmvの実行によって、作業用ファイルの名前

をもとの文書の名前に付け換えられる。

【０１７５】上述したように第３の実施例によれば、指定された構造化文書中の文書部品の属性を指定された属性値に変更することができる。

【０１７６】次に本発明の第４の実施例を図１６を参照して説明する。

【０１７７】図１６は本発明に係る文書処理装置の第４の実施例を機能ブロック図で示したものである。この機能ブロック図は、図１に示した第１の実施例の機能ブロック図の構成において、構成子記述列取得部１５０を削除し、文書名出力部１４００を追加した構成になっている。なお同図において、図１に示した構成要素と同様の機能を果たす部分には同一の符号を付している。

【０１７８】文書名出力部１４００では、文書オペレータ１４１０が、第２の識別子列取得部１４０により得られた識別子列の識別子数を検査し、この結果が指定された第３の条件としての条件情報１４２０に適合する場合は、予め記述された文書の文書名を出力する。なお条件情報１４２０文書名を取得するための条件を示す情報である。

```
#!/bin/sh
(select-type -e Figure $1 ; select-type -e Segment $1) |union |＼
    id-doc -gt 10 $1
```

このスクリプトは、与えられた文書にある図と節の数を調べ、図と節が計１０個以上あったときには、与えられた文書の名前を出力するものである。条件を満たさないときには、何も出力しない。

【０１８３】select-type コマンドとunion コマンドは、第３の実施例のスクリプト例３におけるselect-type コマンド及びunion コマンドと同様の目的で用いられる。id-docコマンドの実行により、union コマンドの実行によって得られた識別子の数を検査し、この検査結果が１０よりも多い場合は、与えられた文書名が標準出力に書き出される。

【０１８４】上述したように第４の実施例によれば、指定された条件に適合する構造化文書の文書名のみを得ることができる。

【０１８５】次に本発明の第５の実施例を図１７を参照して説明する。

【０１８６】図１７は、本発明に係る文書処理装置の第５の実施例を機能ブロック図で示したものである。この機能ブロック図は、第１～第４の実施例の機能ブロック図を合成した構成になっている。すなわち、図１に示した第１の実施例の構成に、図１４に示した第２の実施例の機能ブロック図の構成子記述列挿入部１２００、図１５に示した第３の実施例の機能ブロック図の構成子記述列変更部１３００、図１６に示した第４の実施例の機能ブロック図の文書名出力部１４００を追加した構成になっている。

```
#!/bin/sh
select-type -e Segment $2 | bottom > pid
select-type -e Figure $1  | extract -s - $1  |＼
    insert -yc pid - $2
```

このスクリプトは、二つの文書を入力として、第１引数で指定された文書に含まれる図を、第２引数で指定された文書の最下位の節のうち最初のものの末子として挿入するものである。

【０１９０】実行文の１行目は、第３の実施例のスクリプト例３で説明した通りである。２番目のselect-type コマンドは、第３の実施例のスクリプト例３における最初のselect-type コマンドと同様の目的で使用される。extract コマンドの実行により、第１引数で指定された文書の図を表す構成子記述列が抜き出される。最後に、extract コマンドの実行によって得られた構成子記述列を、insert コマンドの実行によって、第２引数で指定された文書に挿入する。挿入は、第２の実施例のスクリプト例２と全く同様にして行われる。

【０１８７】なおこの第５の実施例では、スクリプトを実行するシェルの処理は、上記第１の実施例の図８乃至図１１で説明した処理と同様である。

【０１８８】次に、実際に処理を行うためのコマンドの仕様例、及びそのコマンドを用いたスクリプトの例を以下に示す。

【０１８９】◇［スクリプト例５］

【０１８０】なおこの第４の実施例では、第１の識別子列取得部１３０の構成子レゾルバ１４１は上述したｓｅｌｅｃｔ－ｔｙｐｅコマンドに相当し、第２の識別子列取得部１４０の識別子オペレータ１４１は上述したｕｎｉｏｎコマンドに相当するように設定されている。またスクリプトを実行するシェルの処理は、上記第１の実施例の図８乃至図１１で説明した処理と同様である。

【０１８１】次に、実際に処理を行うためのコマンドの仕様例、及びそのコマンドを用いたスクリプトの例を以下に示す。

【０１８２】◇［スクリプト例４］

【０１７９】この第４の実施例においても、文書名出力部１４００は、この構成要素の機能を遂行させるためのソフトウェア（プログラム）を制御手段が実行することにより実現される。従って、文書オペレータ１４１０は上述した文書オペレータのコマンド（つまりｉｄ－ｄｏｃ）に相当する。

【０１９１】◇［スクリプト例６］

```
#!/bin/sh
select-type -e Segment $2 | bottom > pid
select-type -e Figure $1  | chval -n Patent  特許 - $1 |＼
    insert -yc pid - $2
```

このスクリプトは、スクリプト例５とほぼ同じ働きをするが、図を表す部分木の根の構成子の属性値を更新するものである。

【０１９２】実行文の１行目は第３の実施例のスクリプト例３で説明した通りである。２番目のselect-type コマンドは第２の実施例のスクリプト例２における最初のselect-type コマンドと同様の目的で使用される。chval コマンドは、第３の実施例のスクリプト例３におけるchval コマンドと同様にして用いられる。最後に、insertコマンドの実行により、上記スクリプト例５におけるinsertコマンドの実行による挿入と同様の処理が行われる。

【０１９３】◇［スクリプト例７］

```
#!/bin/sh
(select-type -e Figure $1 ; select-type -e Segment $1) |union |＼
    extract -s $1 > part
echo -n > empty
chval -n Patent  特許 empty part
```

このスクリプトは、図あるいは節を表す構成子記述列を検索した後、構成子の属性“user visible name ”の値を変更するものである。

【０１９４】select-type コマンド、union コマンド、extract コマンドにより、引数で指定された文書から図あるいは節を表す部分木が検索され、対応する構成子記述列が作業用ファイルpartに格納される。次に、ＵＮＩＸの標準コマンドecho を引数なしで使用することにより、空きのファイルempty が生成される。最後にchval コマンドの実行により、ファイルpartに格納された全構成子記述列を操作対象として、属性"user visible name”の値に出現する文字列“Patent”が“特許”で置換される。

【０１９５】◇［スクリプト例８］

```
#!/bin/sh
select-type -e Segment $2 | bottom > pid
select-type -e Figure $1  > fig
extract -s fig $1 | chval -n Patent  特許 fig $1 |＼
    insert -yc pid - $2
```

このスクリプトは、上記スクリプト例７の例とほぼ同じ働きをする。なお、このスクリプト例８においては、最終的に挿入を行う位置を、上記スクリプト例６の例と同様に決定している点が、上記スクリプト例７と異なる。

【０１９６】◇［スクリプト例９＝テキスト処理機能との組み合わせの例］

```
#!/bin/sh
select-type -e Figure $1| extract -s - $1  |＼
    sed -e "user-visibe-name/s/Patent/特許/g"
```

この例では、テキスト操作を行うＵＮＩＸの標準コマンドｓｅｄを用いて、検索結果得られた構成子記述列中の属性“user visible name ”の値に含まれる文字列“Patent”を“特許”に置換するものである。

【０１９７】次に、図１２に示した図の論理構造を有する文書に対して処理を行う場合について説明する。

【０１９８】ある文書から図だけを検索して、それらだけから文書を生成するためのスクリプトの例を以下に示す。

【０１９９】◇［スクリプト例１０］

```
%  select-type -e  Figure test.doc |extract -s - test.doc |＼
    insert -yc -p "3" - skeleton.doc  > result.doc
```

最初に、select-type コマンドの実行により、test.docなるファイルから、属性“uobject-type”の値に部分文字列“Figure”を含む構成子の識別子列を列挙し、標準出力に書き出される。次に、extract コマンドの実行により、標準入力から識別子列が読み込まれ、各識別子を根とする部分木に対応する構成子記述列がファイルtest.docから抜き出される。最後にinsertコマンドの実行により、抜き出された部分木が標準入力から読み込まれ、この部分木がファイルskeleton.docに繋ぎ込まれ、こうして得られた構成子記述列が標準出力に書き出される。ここで、ファイルskeleton.docは文書のヘッダ部分と特定論理根だけからなる。結果として得られた構成子記述列は、result.docなるファイルにリダイレクトされ、出力文書となる。

【０２００】次に、コマンドプロシジャ例９の修正版で、見出しに"viewpoint” という文字列を含む図を検索

して、そのような図だけからなる文書を生成するためのスクリプト例を以下に示す。

```
%   (select-type Figure test.doc ; select-content＼
     viewpoint test.doc)  | above | extract -s - test.doc | ＼
     insert -yc -p  "3" - skeleton.doc ＞ result.doc
```

ファイルskeletonは上記スクリプト例１０と同様である。この例では、select-contentコマンドは、"viewpoint" という文字列を含む内容部の識別子を列挙するために用いられる。above コマンドの実行により、２つの識別子列が読み込まれ、述語above が成立する構成子の識別子列が出力される。各識別子列は構成子レゾルバ（select-type 、select-content）によって生成されたものであり、従ってある述語が成立する構成子の識別子列である。従ってabove コマンドの実行結果（出力）は見出しに"viewpoint" という文字列を含む図の識別子列である。

【０２０２】次に、見出しに “introduction” という文字列を含む章（或いは節）に含まれる最下位の節を選び、それらを一つの文書にまとめるためのスクリプト例を以下に示す。

【０２０３】◇［スクリプト例１２］

```
#!/bin/sh
(select-type -e   Segment test.doc ; ＼
   (select-content -p introduction test.doc  ; ＼
  select-type -e title test.doc) | ＼
   below)  | ＼
  above  | ＼
  bottom | extract -s - test.doc  | ＼
  insert -yc -p "3" - skeleton.doc  ＞ result.doc
```

below コマンドは２つの構成子記述列を読み込み、述語below が成立する構成子の識別子列を出力する。従って、この例では、属性 “uobject-type” の値が文字列"title" を含む構成子に従属する内容部のうち、introductionという文字列を含むものを列挙する。

【０２０４】bottomコマンドは、一つの識別子列を読み込み、述語bottomが成立する構成子記述列を出力する。

【０２０５】上記各実施例では、文書名を得る方法として、（ａ）文書名をスクリプトに記述する。（ｂ）スクリプトに実行する引数として文書名を与える。の２通りの方法があるが、これを、構成子記述列から得るようにしても良い。

【０２０６】すなわち、図１７中点線で示されるように、文書名列取得部１５１０と、文書名列保持部１５２０を設ける。

【０２０７】文書名列取得部１５１０においては、文書レゾルバ１５１１が、文書名列を取得するための条件を示す情報である指定された条件情報１５１２に基づいて、構成子記述列保存部１１０から文書名（単数又は複数）を取得し、この文書名を文書名列保持部１５２０に格納する。

【０２０８】そして文書名出力部１４００が、識別子列保持部１２０に保持されている第２の識別子列取得部１４０により得られた識別子列の識別子数を検査し、この結果が指定された第３の条件に適合する場合は、文書名列保持部１５２０に保持されている文書名を出力する。

【０２０９】上述したように第５の実施例によれば、指定された構造化文書中から指定された条件に適合する文書部品を検索し、この文書部品を他の構造化文書中に挿入することができる。

【０２１０】また、指定された構造化文書中から指定された条件に適合する文書部品を検索し、この文書部品の属性を指定された属性値に変更することができる。

【０２１１】さらに、指定された構造化文書中から指定された条件に適合する文書部品を検索し、この文書文品の属性を指定された属性値に変更し、更に、この文書部品を指定された構造化文書中に挿入することができる。

【０２１２】上述した各実施例によれば、構成子記述列、識別子列、文書名列のいずれかを扱える文書処理機能（装置）であれば、上述した様な各文書処理を実行することができる。

【０２１３】

【発明の効果】以上説明したように本発明によれば、予め記述された単数又は複数の文書に対応し、且つ、識別子が付与されている複数の構成子を有する構成子記述列から、指定された第１の条件に適合する構成子に対応する識別子列を取得し、この識別子列から、指定された第２の条件に適合する識別子列を取得し、この第２の条件に適合する識別子列に基づいて、前記予め記述された単数又は複数の文書に対応する構成子記述列に対する所定の処理を実行するようにし、この所定の処理をするに際し、構成子記述列の取得（検索）、挿入、構成子記述列の属性値の変更、文書名の出力、の各処理のうち、いずれかの処理又は複数の処理を実行するようにしたので、指定された文書から条件に適合する文書部品を検索することができるという利点がある。

【０２１４】また、ある文書中の文書部品を抽出し、この文書部品を、他の文書中に挿入することができるとい

う利点がある。

【０２１５】また、指定された条件に適合する文書の文書名のみを得ることができるという利点がある。

【０２１６】また、指定された文書中から指定された条件に適合する文書部品を検索し、この文書部品を、他の文書中に挿入することができるという利点がある。

【０２１７】また、指定された文書中から指定された条件に適合する文書部品を検索し、この文書部品の属性を、指定された属性値に変更することができるという利点がある。

【０２１８】また、指定された構造化文書中から指定された条件に適合する文書部品を検索し、この文書文品の属性を指定された属性値に変更し、更に、この文書部品を指定された構造化文書中に挿入することができるという利点がある。

【０２１９】さらに、構成子記述列、識別子列に基づいて文書処理を行うようにしているので、これらの情報を扱える他の文書処理機能と込み合わせて、文書あるいは文書部品の検索、挿入、合成、更新を正確に処理することのできる文書処理方法および文書処理装置を提供することができるという効果を奏する。

【図面の簡単な説明】

【図１】本発明に係る文書処理装置の第１の実施例を示す機能ブロック図。

【図２】図１に示した実施例の装置を実現するためのハードウェア構成を示すブロック図。

【図３】ネットワーク構造の一例を示す図。

【図４】構造化文書における論理構造の一例を示す図。

【図５】図４に示す木構造の各ノードにＤｅｗｅｙの１０進分類法に従ってアドレスを割り当てた様子を示す。

【図６】共通の祖先のうち、最も若いものを求めるための処理手順を示すフローチャート。

【図７】図４に示す木構造の各ノードに行きがけ順に辿ったときに出現する順番に従って番号を割り当てた様子を示す図。

【図８】全体の処理の流れを示すメインルーチン。

【図９】全体の処理において実行されるスクリプトの実行処理を示すサブルーチン。

【図１０】スクリプトの実行処理において実行される基本単位の処理を示すサブルーチン。

【図１１】実行単位の処理において実行される基本実行単位の処理を示すサブルーチン。

【図１２】構造化文書における図の論理構造の一例を示す図。

【図１３】構造化文書における章・節の論理構造の一例を示す図。

【図１４】本発明に係る文書処理装置の第２の実施例を示す機能ブロック図。

【図１５】本発明に係る文書処理装置の第３の実施例を示す機能ブロック図。

【図１６】本発明に係る文書処理装置の第４の実施例を示す機能ブロック図。

【図１７】本発明に係る文書処理装置の第５の実施例を示す機能ブロック図。

【図１８】従来の文書処理を説明するために使用した構造化文書の論理構造を示す図。

【図１９】図１８に示す構造化文書に対する処理結果としての出力文書を示す図。

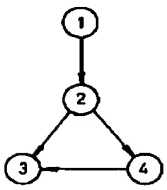
【図２０】従来の文書処理を説明するために使用した構造化文書の論理構造を示す図。

【図２１】図２０に示す構造化文書に対する処理結果としての出力文書を示す図。

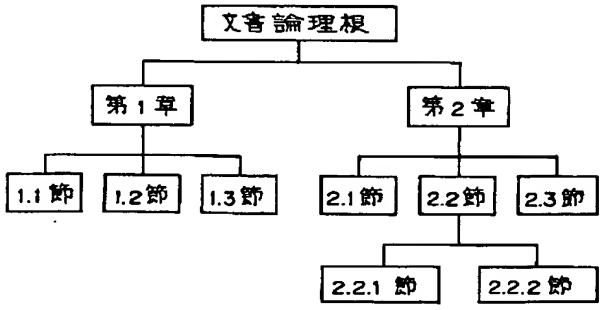
【符号の説明】

１１０…構成子記述列保持部、１２０…識別子列保持部、１３０…第１の識別子列取得部、１４０…第２の識別子列取得部、１５０…構成子記述列取得部、２１０…入力装置、２２０…ディスプレイ、２３０…ディスク、２４０…主メモリ、２５０…中央処理装置、１２００…構成子記述列挿入部、１３００…構成子記述列変更部、１４００…文書名出力部、１５１０…文書名列取得部、１５２０…文書名列取得部。

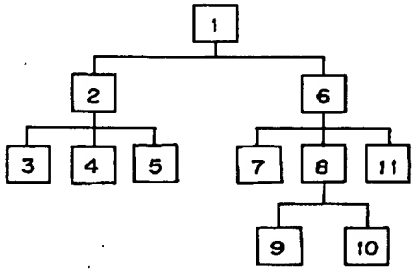
【図３】



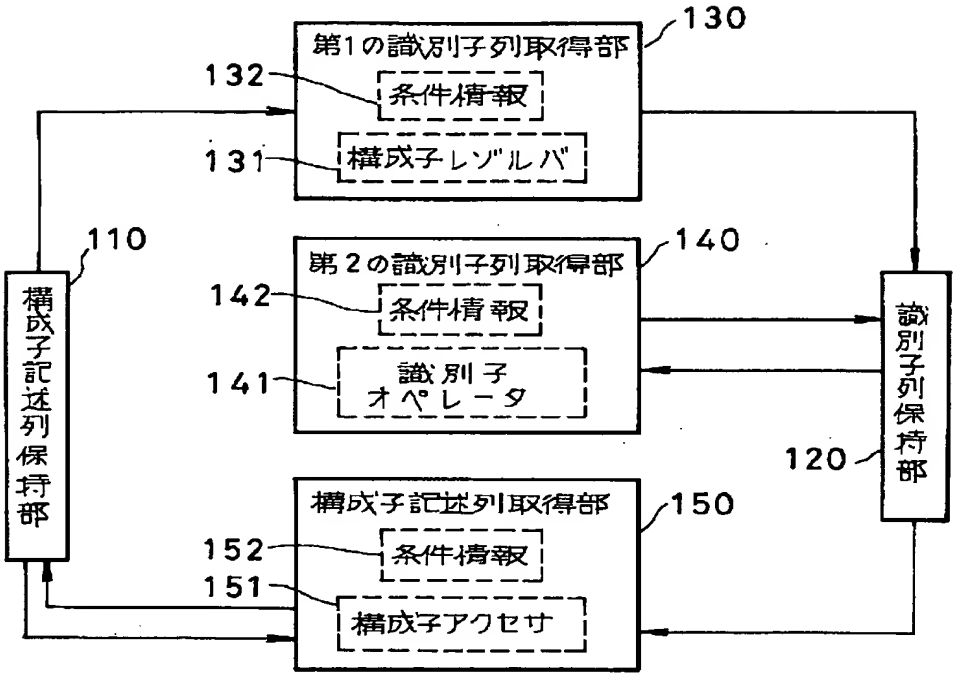
【図４】



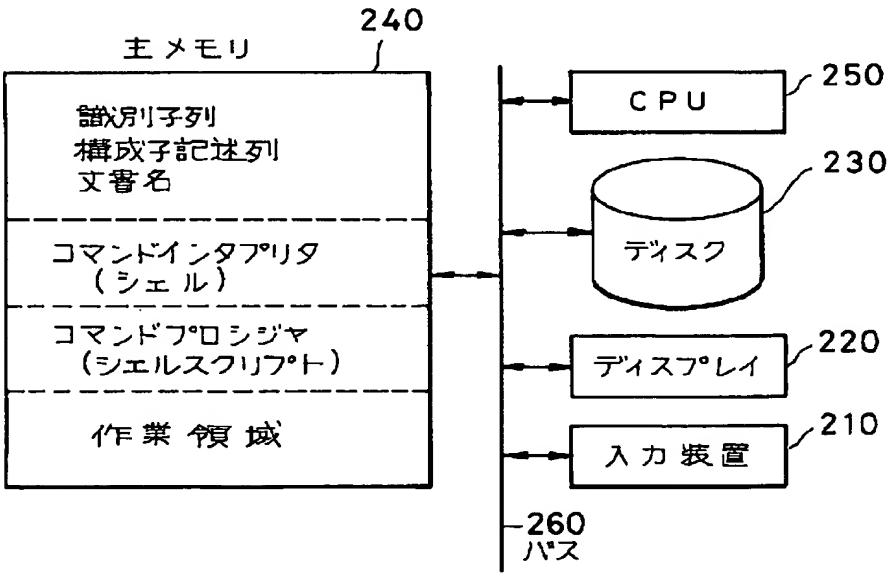
【図７】



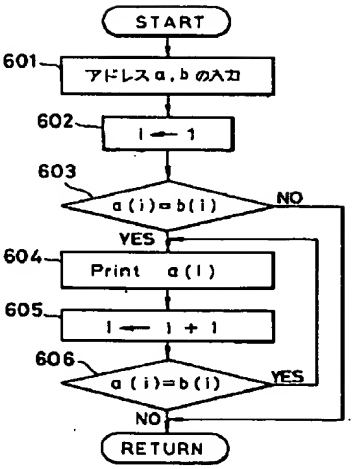
【図１】


第1の識別子列取得部
130
132
条件情報
131
構成子レゾルバ
110
構成子記述列保持部
第２の識別子列取得部
140
142
条件情報
141
識別子
オペレータ
識別子列保持部
120
構成子記述列取得部
150
152
条件情報
151
構成子アクセサ

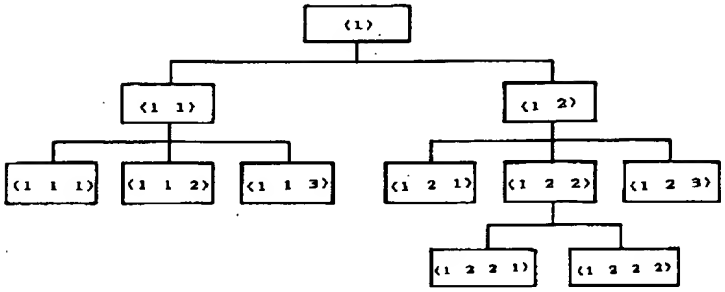
【図２】


主メモリ
240
識別子列
構成子記述列
文書名
コマンドインタプリタ
（シェル）
コマンドプロシジャ
（シェルスクリプト）
作業領域
CPU
250
ディスク
230
ディスプレイ
220
入力装置
210
260
バス

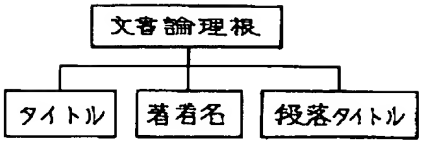
【図６】


START
601
アドレスa,bの入力
602
i ← 1
603
a(i)=b(i)
NO
YES
604
Print a(i)
605
i ← i + 1
606
a(i)=b(i)
YES
NO
RETURN

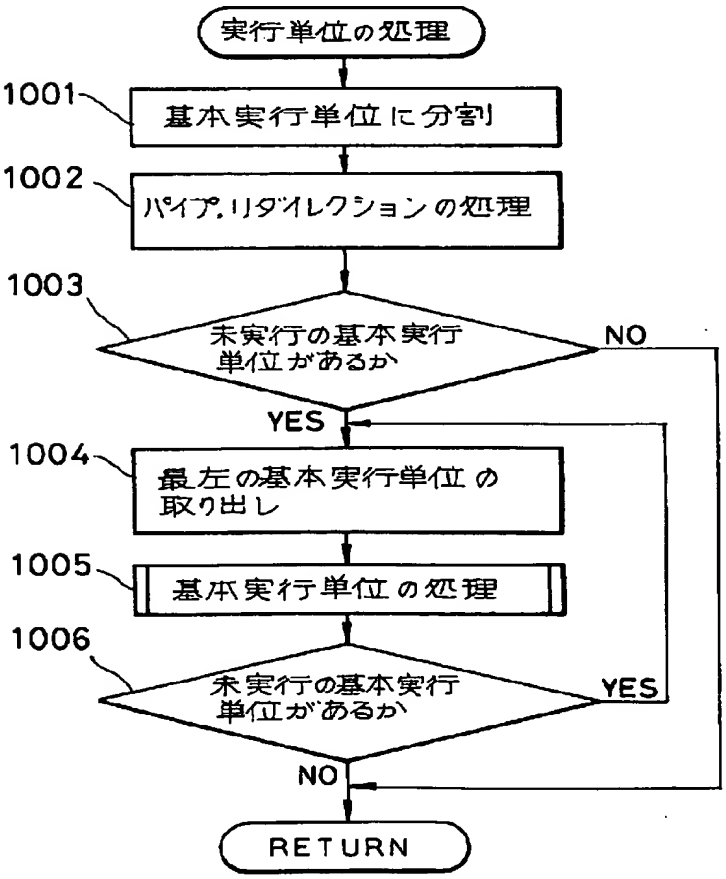
【図５】



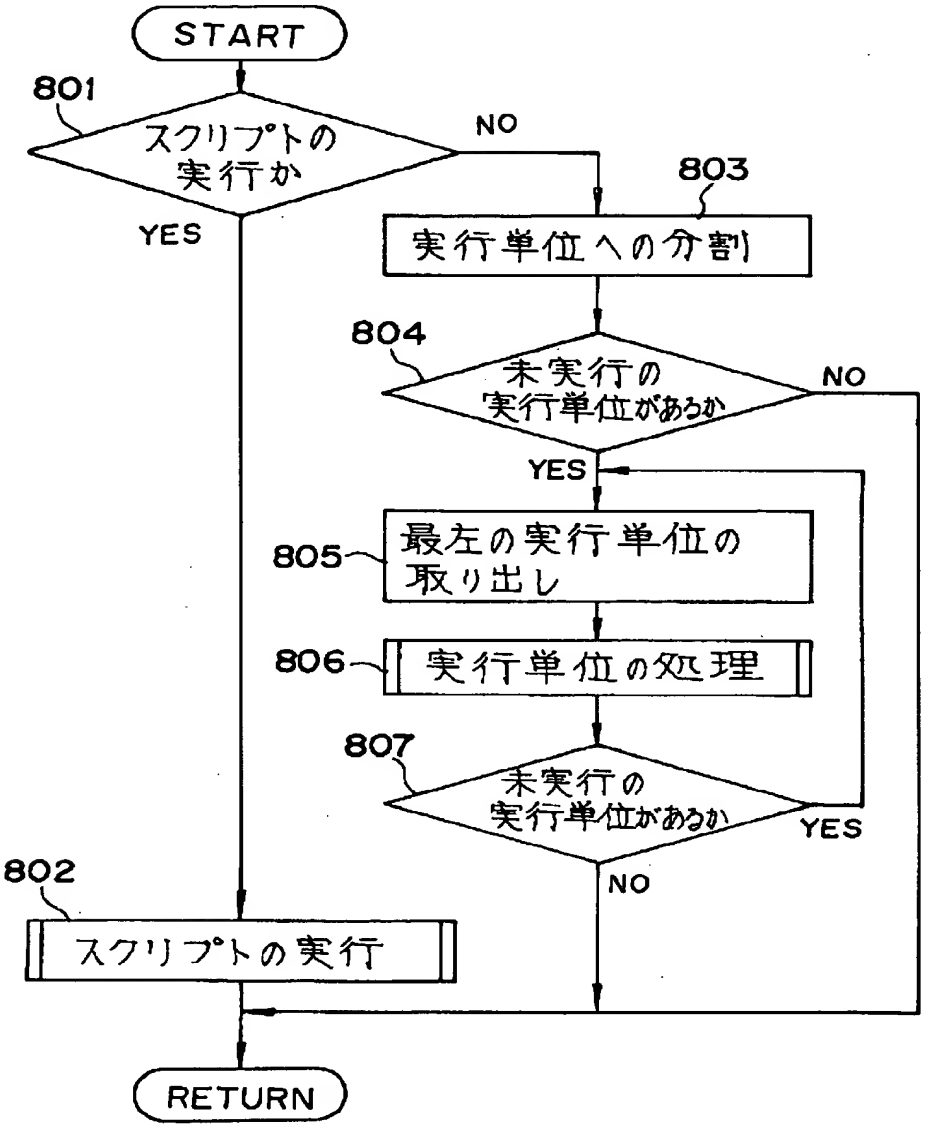
【図２１】



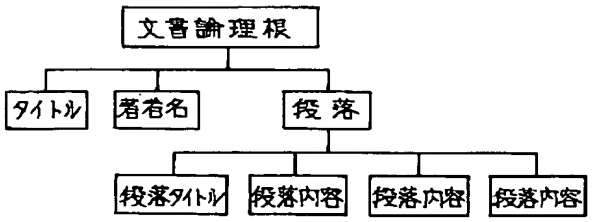
【図１０】



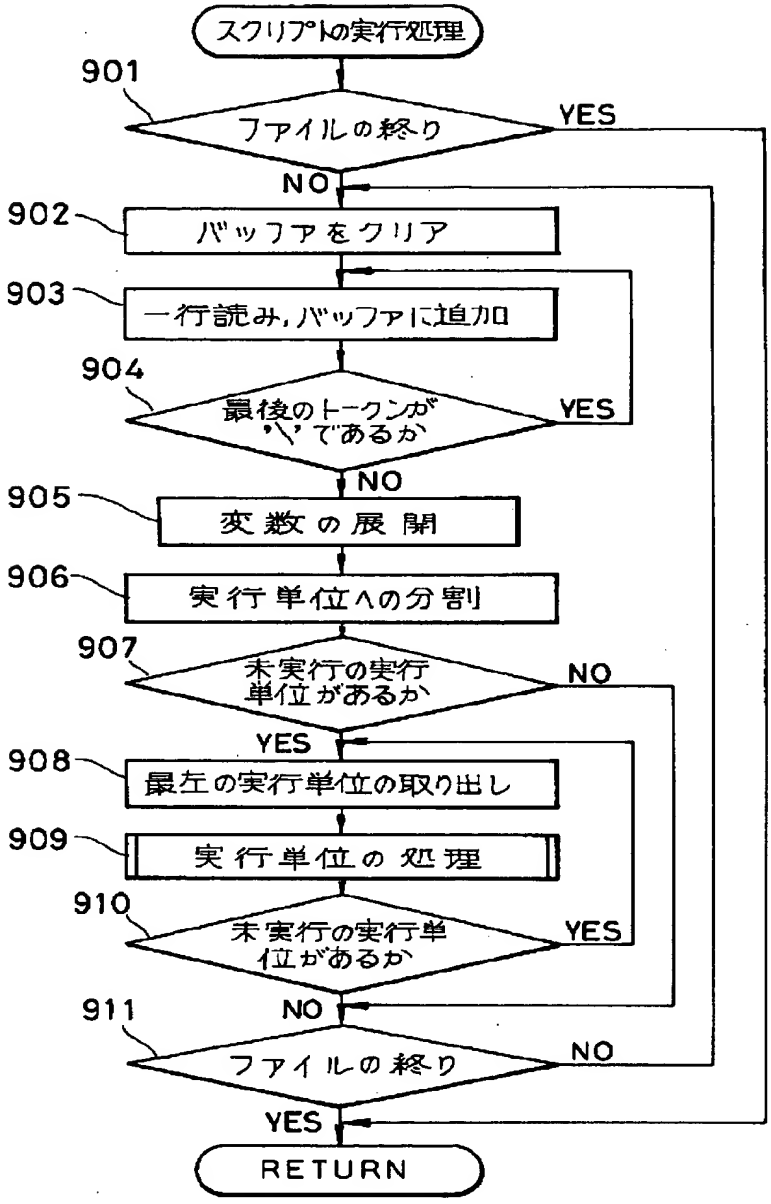
【図８】



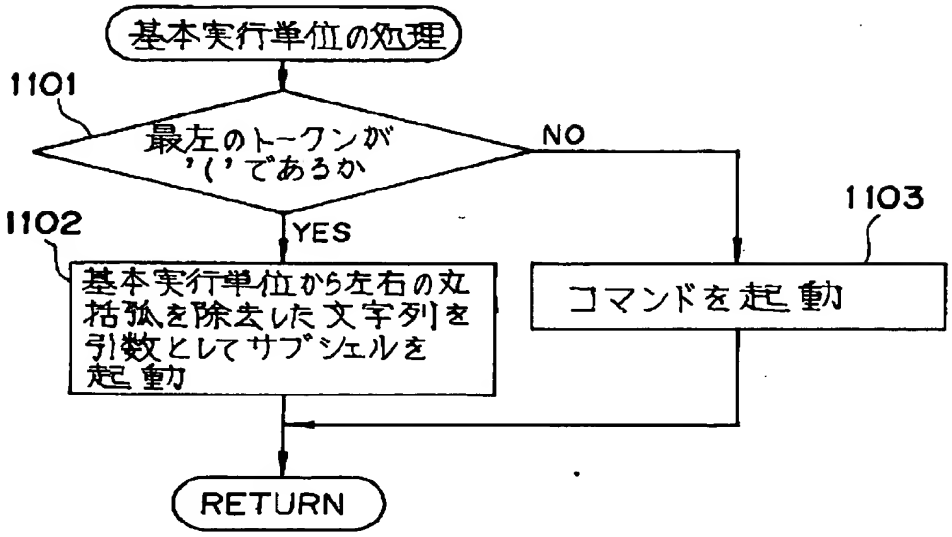
【図２０】



【図９】



【図１１】


基本実行単位の処理
1101
最左のトークンが
'('であるか
NO
YES
1102
基本実行単位から左右の丸括弧を除去した文字列を引数としてサブシェルを起動
1103
コマンドを起動
RETURN

【図１２】

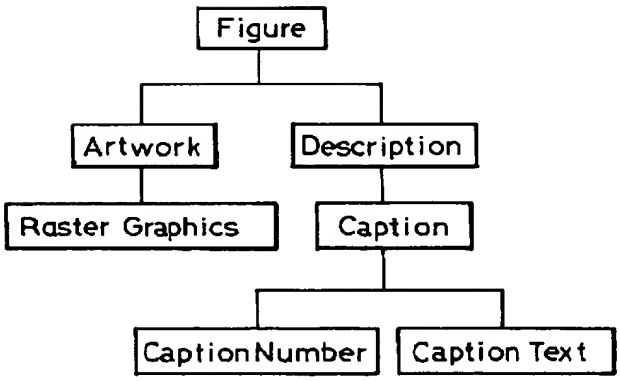
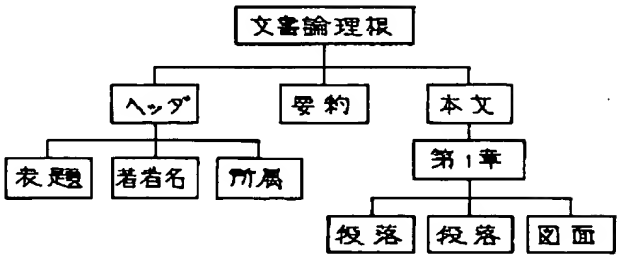
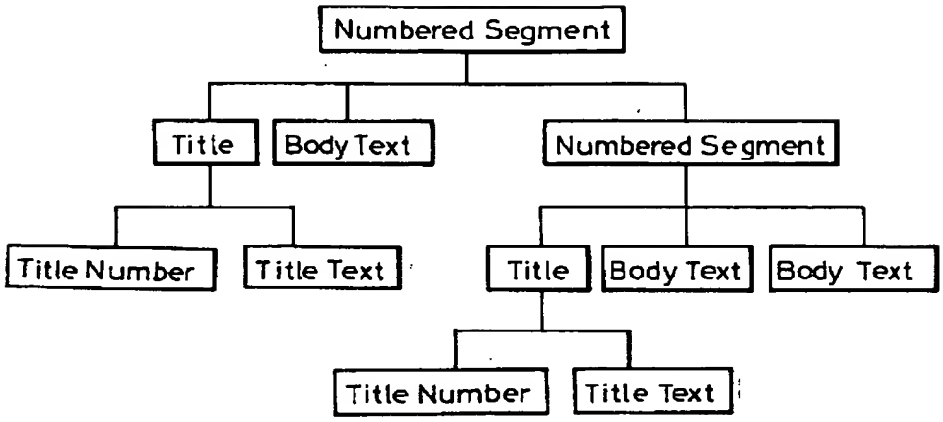


Figure
Artwork
Description
Raster Graphics
Caption
CaptionNumber
Caption Text

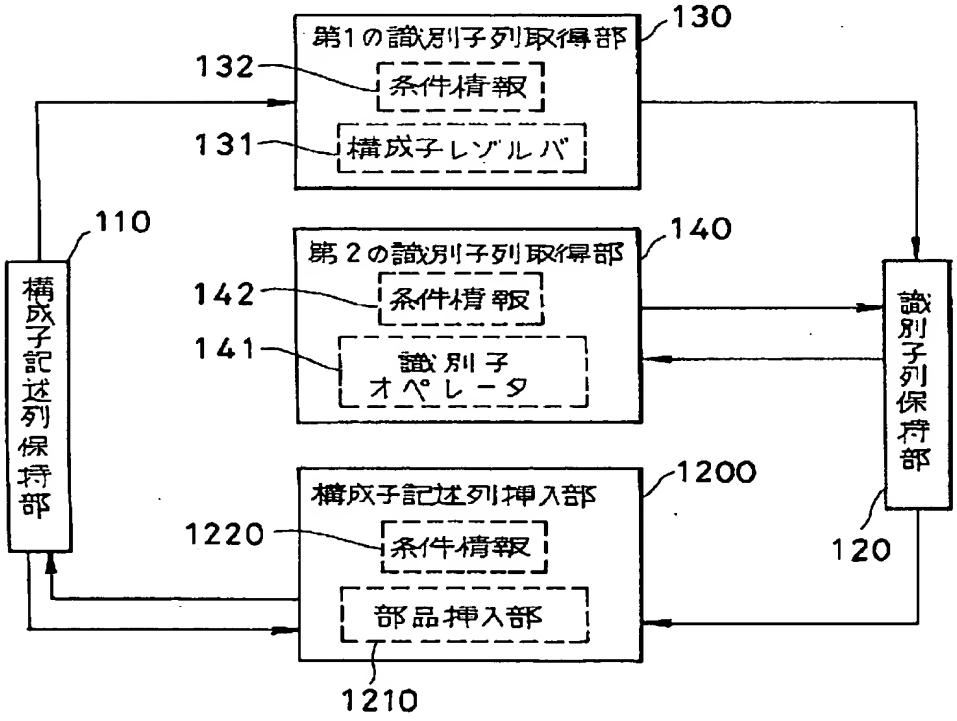
【図１８】


文書論理根
ヘッダ
要約
本文
表題
著者名
所属
第1章
段落
段落
図面

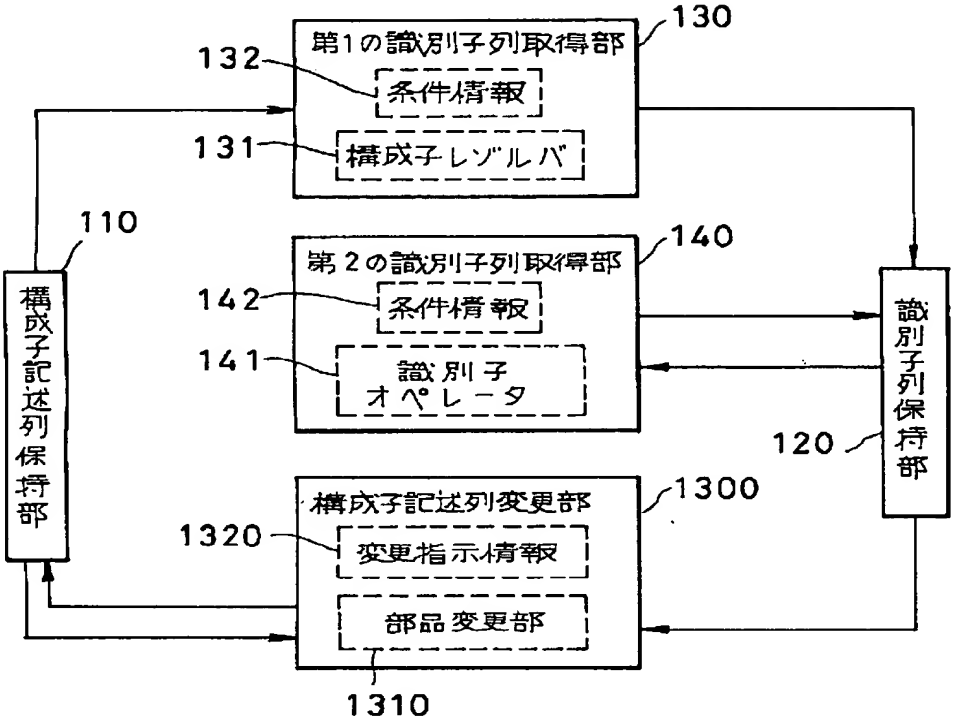
【図１３】


Numbered Segment
Title
Body Text
Numbered Segment
Title Number
Title Text
Title
Body Text
Body Text
Title Number
Title Text

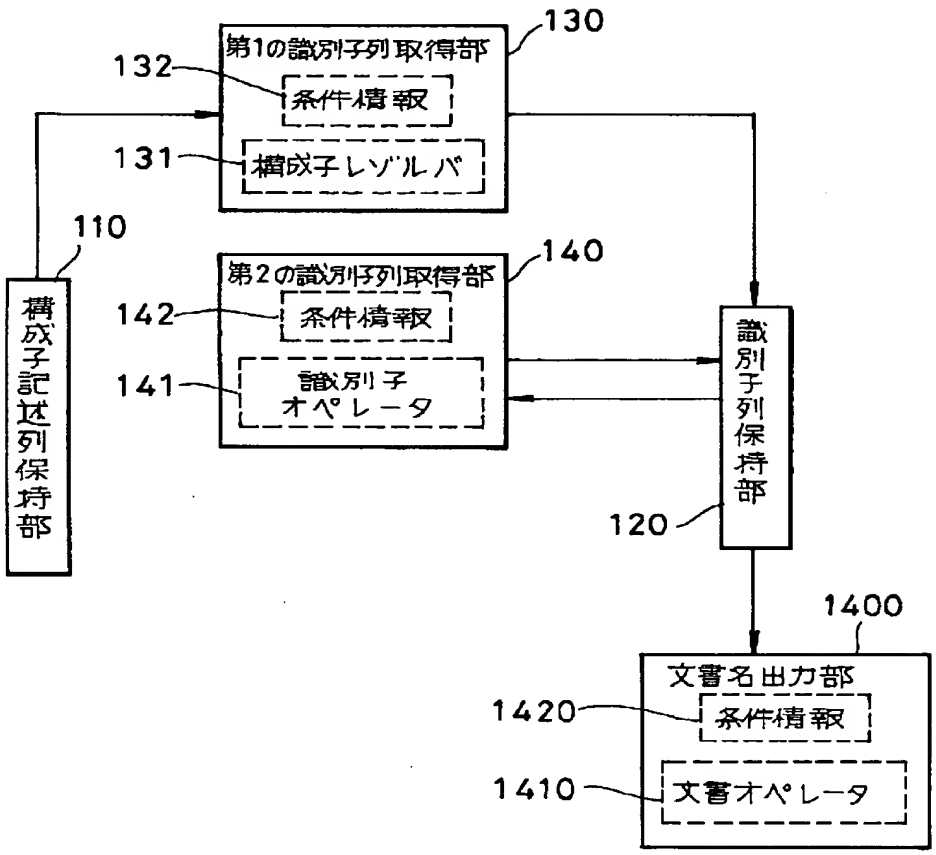
【図１４】


第1の識別子列取得部
130
132
条件情報
131
構成子レゾルバ
110
構成子記述列保持部
第２の識別子列取得部
140
142
条件情報
141
識別子
オペレータ
識別子列保持部
120
構成子記述列挿入部
1200
1220
条件情報
部品挿入部
1210

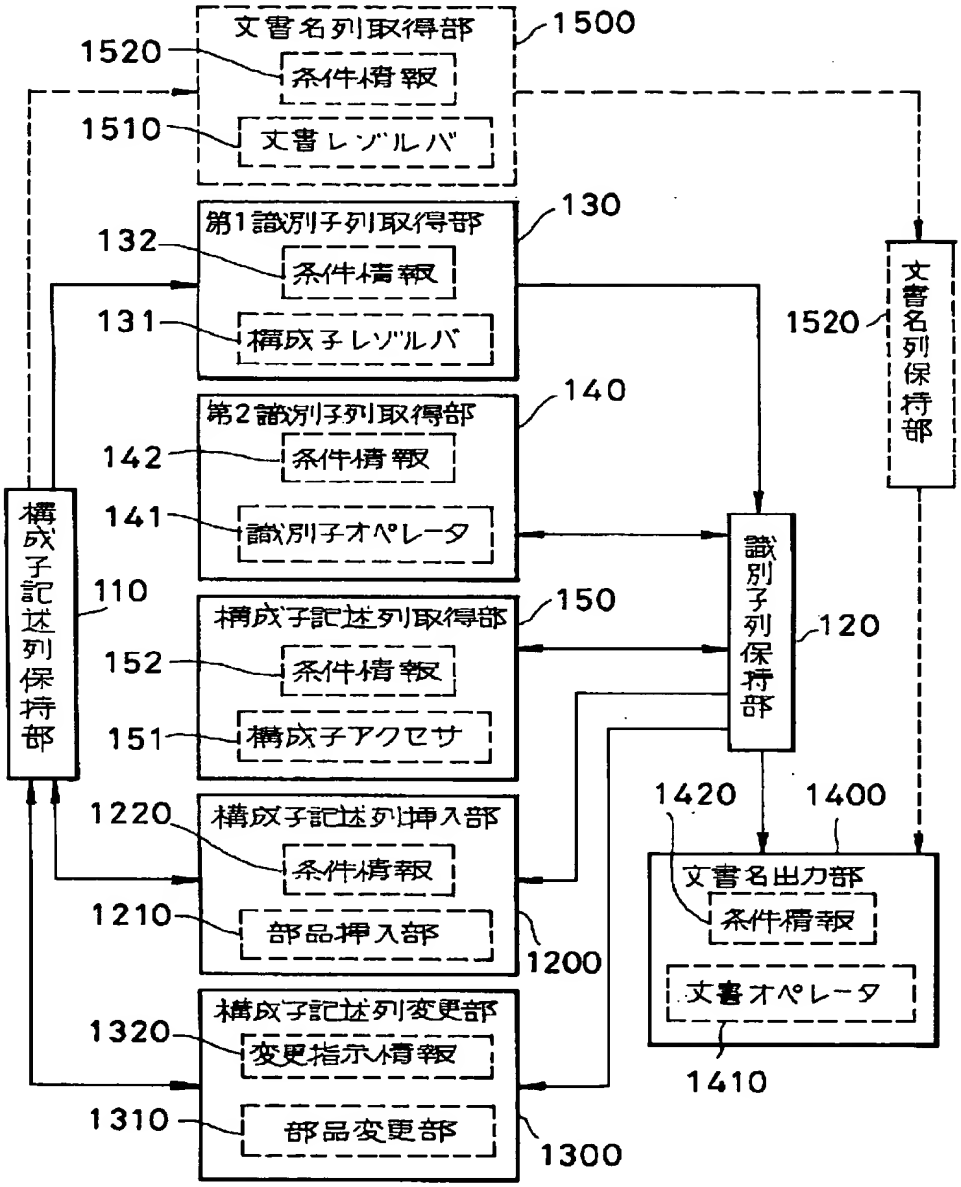
【図１５】



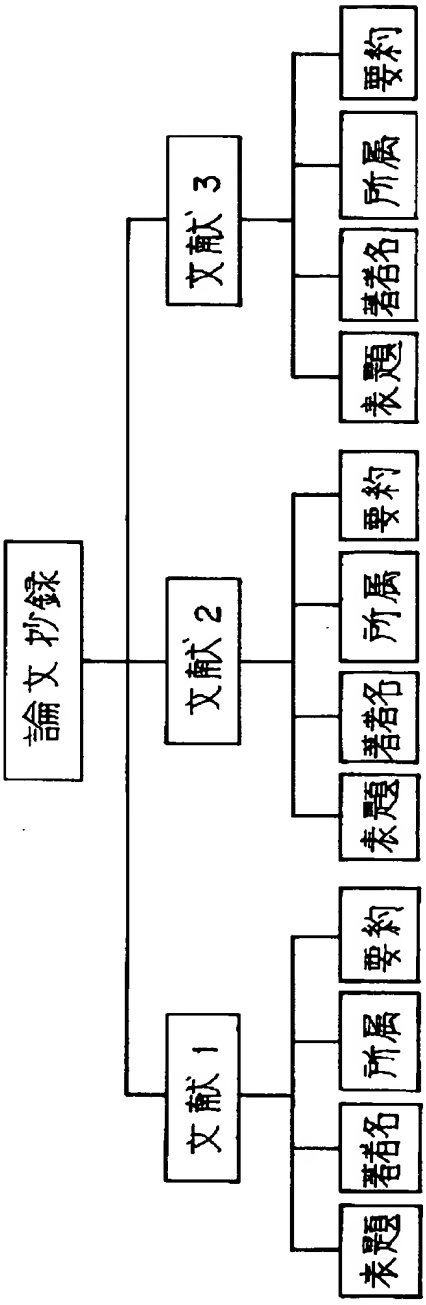
【図１６】



【図１７】



【図１９】



【手続補正書】
【提出日】平成５年３月１９日
【手続補正１】
【補正対象書類名】明細書
【補正対象項目名】０１２０
　　ｐｒｅｄ（ａ，ｂ）⇔$a_{length(lub(a,b))+1} \leqq b_{length(lub(a,b))+1}$
なおｐｒｅｄ（ａ，ｂ）はａ、ｂのうち行きがけ順で先に表されるものを求める関数である。
【手続補正２】
【補正対象書類名】明細書
【補正対象項目名】０１２１
　　ｓｕｃｃ（ａ，ｂ）⇔$a_{length(lub(a,b))+1} \geqq b_{length(lub(a,b))+1}$
なおｓｕｃｃ（ａ，ｂ）はａ、ｂのうち行きがけ順で後に表されるものを求める関数である。
【補正方法】変更
【補正内容】
【０１２０】ここで述語ｐｒｅｄを次のように定義する。
【補正方法】変更
【補正内容】
【０１２１】同様に、述語ｓｕｃｃを次の様に定義する。

フロントページの続き

(72)発明者　村田　真
神奈川県横浜市保土ケ谷区神戸町134番地
横浜ビジネスパーク　イーストタワー
富士ゼロックス株式会社内